ROUGH DRAFT

SANDIA REPORT

SAND# • UC-???
Classification
Printed 4/14/97

MUSTAFA User's Guide

Version 1.0

Micheal W. Glass

Inside of Cover Page

ROUGH DRAFT

SAND #
Classification
Printed ????

Distribution
Category UC-???

MUSTAFA User's Guide

Version 1.0

Micheal W. Glass
Incompressible Fluid Dynamics Department
Sandia National Laboratories
Albuquerque, New Mexico 87185

Abstract

MUSTAFA is a Motif GUI driven, full-featured data visualizer for unstructured grid data in the EXODUS II database format. MUSTAFA is built with the Express application development environment (AVS Inc.) which makes it fairly easy to modify and extend to support other data formats, special visualization methods, etc. MUSTAFA supports both nodal and element based data.

MUSTAFA has four main components: (1) data readers, which actually read the data from disk and set up data structures; (2) data filters, which only manipulate the data or mesh; (3) visualization methods, which actually produces a viewable object; and (3) viewers, which display the results of the visualization methods onto the computer screen and provide for user specified geometric transformations. Multiple simultaneous data readers and viewers are supported. MUSTAFA's visualization tools are divided into four primary groups depending upon what kind of data they work on:

1) mesh visualization tools: these tools require no data, just the mesh structure. Tools are available to view the mesh, element blocks, side sets, and node sets.
2) scalar visualization tools: these tools work on scalar data. Tools presently available for scalar data are paint cells, multi-material map, external edges, external faces, slice plane, iso lines, iso surface, and iso volume.
3) vector visualization tools: these tools work on vector data. Since the EXODUS II database specification does not support vector style data, each of the tools in this category have a vector construction popup control panel which lets the user define which of the scalar components in the database make up the individual vector components. Tools presently available for vector data are hedge hog, streamlines, and particle advection.
4) annotation tools: These tools don't really work on any data. They allow the user to add simple annotations to the viewer. These tools include the EXODUS II time value, the SNL logo, and any user text.

ROUGH DRAFT

Table of Contents

List of Figures

1. Introduction

MUSTAFA is a scientific visualization tool for unstructured grid data stored in the EXODUS II database format. MUSTAFA has been around the Engineering Sciences Center in one form or another for a number of years though it has never received any funding, hence the acronym. I just decided that as one of the *"crown jewels"* of the national laboratory system, we were very weak in the area of visualization tools and something had to be done instead of continually waiting for the grand unified visualization environment of the future (whatever that means!). In its latest incarnation, MUSTAFA is a Motif GUI driven, full-featured scientific data visualizer built with the Express application development environment (AVS Inc.). If you do decide to use MUSTAFA and notice bugs, lack of features, etc., just let me know. I can't promise everything will get fixed or addressed right away but I will try. Since MUSTAFA was developed on my own time (late nights, weekends, etc.), it's not really a supported package (but then again, when has code maintenance ever been an issue?). On the plus side, since MUSTAFA was built with Express, it is fairly easy for me to maintain and modify.

As previously mentioned, MUSTAFA was primarily written to support EXODUS II databases but it can easily be extended to support other database formats. Both nodal and element data, along with global data is supported. Presently there are only two visualization methods that directly support element data - a "paint cells" method and a "multi-material map" method (along the lines of CTHplot). By default, element data is also interpolated to the nodes so all the node based methods (e.g. isosurface, slice, cut, etc.) can be also be used on element data. For element data that has been interpolated to the nodes, an asterisk (*) precedes the variable name in all the data component selection boxes.

MUSTAFA works on data with four main components: (1) data readers, which actually read the data from disk and set up data structures; (2) data filters, which only manipulate the data or mesh; (3) visualization methods, which actually produces a viewable object; and (3) viewers, which display the results of the visualization methods onto the computer screen and provide for user specified geometric transformations. Data filters currently available include: (1) mesh displacements; (2) mirroring; (3) element block selection; (4) data thresholding; (5) and data clamping. MUSTAFA's visualization tools are divided into four primary groups depending upon what kind of data they work on:

1. mesh visualization tools: these tools require no data, just the mesh structure. Tools are available to view the mesh, element blocks, side sets, and node sets.
2. scalar visualization tools: these tools work on scalar data. Tools presently available for scalar data are paint cells, multi-material map, external edges, external faces, slice plane, iso lines, iso surface, and iso volume.
3. vector visualization tools: these tools work on vector data. Since the EXODUS II specification does not support vector style data, each of the tools in this category have a vector construction popup control panel which lets the user define which of the scalar components in the database make up the individual vector components. Tools presently available for vector data are hedge hog, streamlines, and particle advection.
4. annotation tools: These tools don't really work on any data. They allow the user to add simple annotations to the viewer. These tools include the EXODUS II time value, the SNL logo, and any user text.

Two forms of output are presently supported, rasterfile output and printfile output. As one might suspect, rasterfile output creates a rasterfile (in a variety of formats) of the active viewer. Rasterfiles are useful for importing images into word processing documents (e.g. Frame, Word, etc.) or web pages, or for encoding into mpeg movie sequences. Printfile is used to create postscript (including eps) or cgm files that can sent to a printer for printing.

1.1 Availability

MUSTAFA is currently available on the Bldg. 880 1500 LAN and on the Engineering Sciences Silicon Graphics Visualization Sever (hostname sasg249 or bedrock2). On the Bldg. 880 1500 LAN, MUSTAFA can be run locally on your desktop workstation or on the SparcCenter compute server (sass830). To run MUSTAFA, simply type `mustafa` on the command line. There are currently five (5) floating runtime licenses for Express on the LAN. To run MUSTAFA on the visualization server, type the command `~mwglass/etc/mustafa`. There are currently five (5) floating runtime licenses for Express on the visualization server. When running MUSTAFA locally, it will attempt to use any hardware accelerator present on your system. Those of you with Sparc 20 workstations will notice some hardware graphics acceleration with the SX frame buffer. When running on a remote machine(e.g. bedrock2 or sass830), the software renderer will be utilized and performance will depend upon cpu utilization and network traffic. Though the Express environment is supported on a variety of hardware platforms, MUSTAFA is currently limited to the platforms to which I have access. Binary versions of MUSTAFA can be generated and distributed for Sun Solaris 2.x, HP HPUX, and SGI IRIX systems.

1.2 Comments on the GUI

As mentioned earlier, the GUI is Motif based so interaction is the same as with normal Motif widgets. For typein widgets, the RETURN key must be entered for the new value to take effect. At some places in the application, there are slider widgets with a small box to the right with a “?” displayed in it. Clicking on this box will popup a dialog box with a typein widget that can be used to precisely set the slider value.

2. Overview

2.1 Basic Overview

MUSTAFA consists of three elementary building blocks: (1) Data Groups; (2) Visualization Groups; and (3) Viewers. Multiple instances of these three base objects can exist at any time as shown in Figure 2.1. A Data Group is further subdivided into (1) Data Readers and (2) Global Filters. There can be

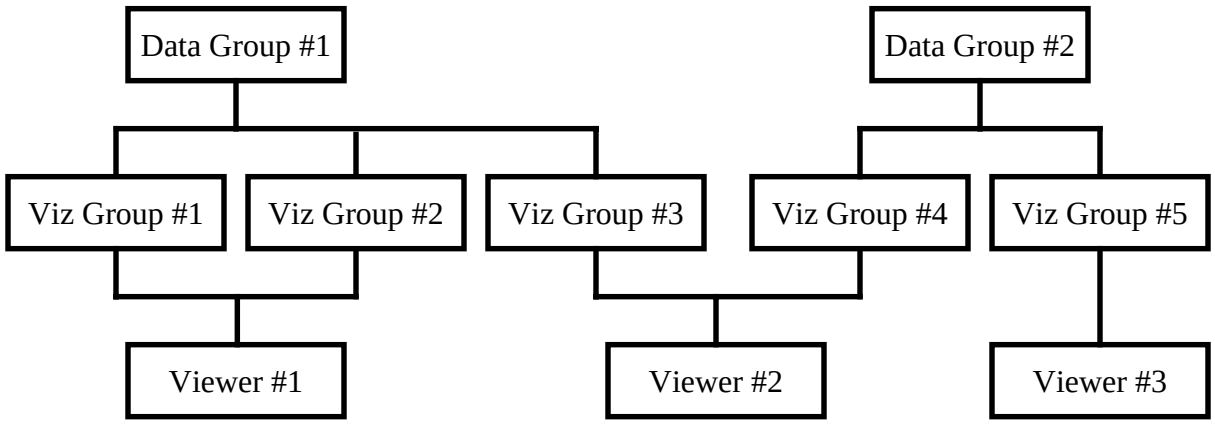


Figure 2.1 Sample network layout

multiple readers for various data formats within the data reader object. The global filters operate serially on the data from the data reader and its output is used as the input for all visualization groups attached to that particular data group. Each visualization group is subdivided into (1) Local Filters and (2) Viz Method. The local filters operate serially on only the input data to a particular visualization group. This layout allows for a fairly flexible visualization environment. Perhaps the most useful “Local Filter” is the element block selection filter which allows each visualization method to operate on a different set of element blocks. The viz method itself produces a particular renderable visualization of the data. The viewers display the 2- or 3-D renderable objects on the screen. Each viewer is a separate window and may be moved and resized at will. An expanded view of MUSTAFA is shown in Figure 2.2

2.2 Startup

Upon startup, MUSTAFA will create three X-windows as shown in Figure 2.3. The rightmost window is the default viewer and the windows on the left are the main menu window and the main control window. Access to the GUIs for the data groups and visualization groups are via the main menu window.

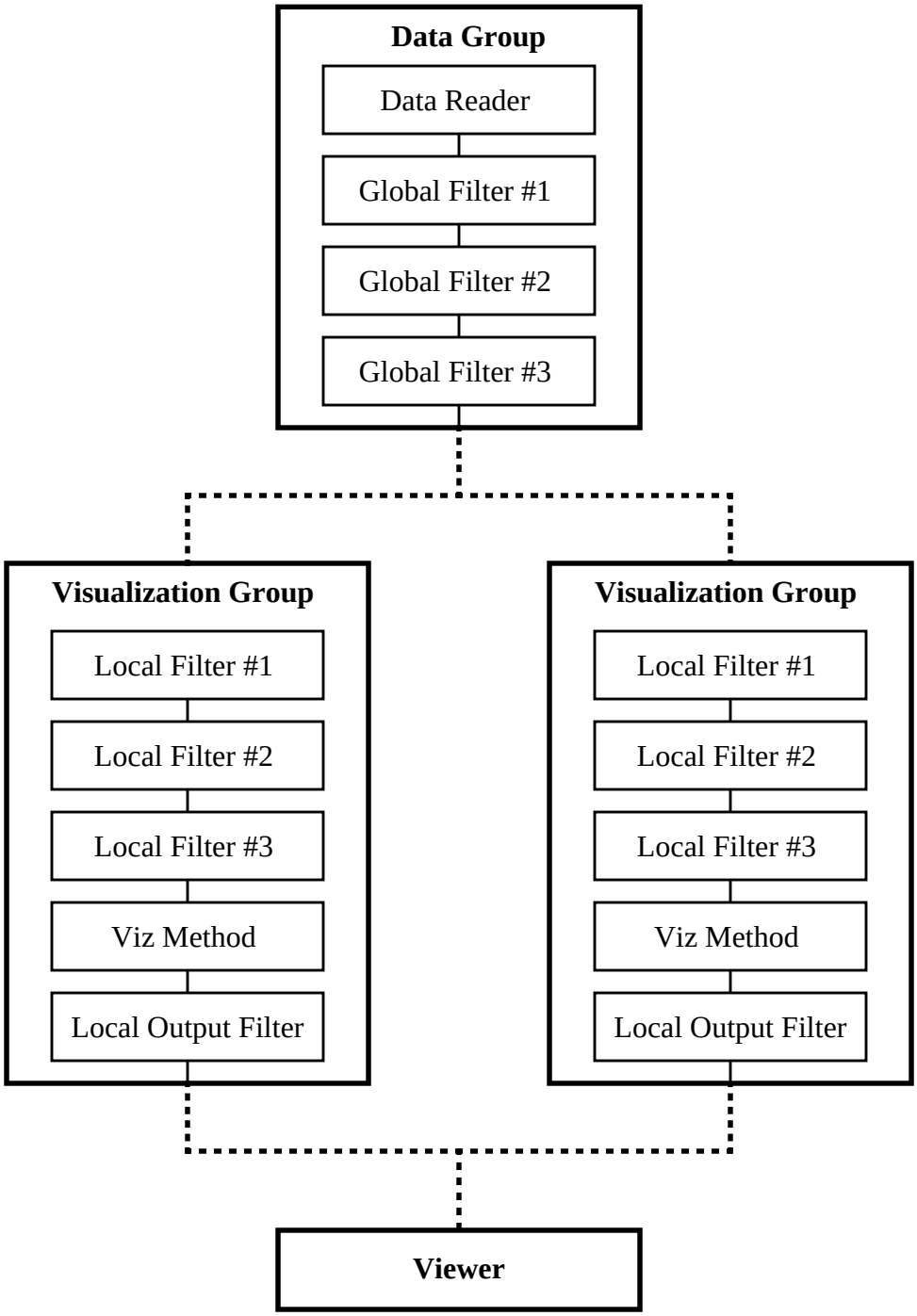


Figure 2.2

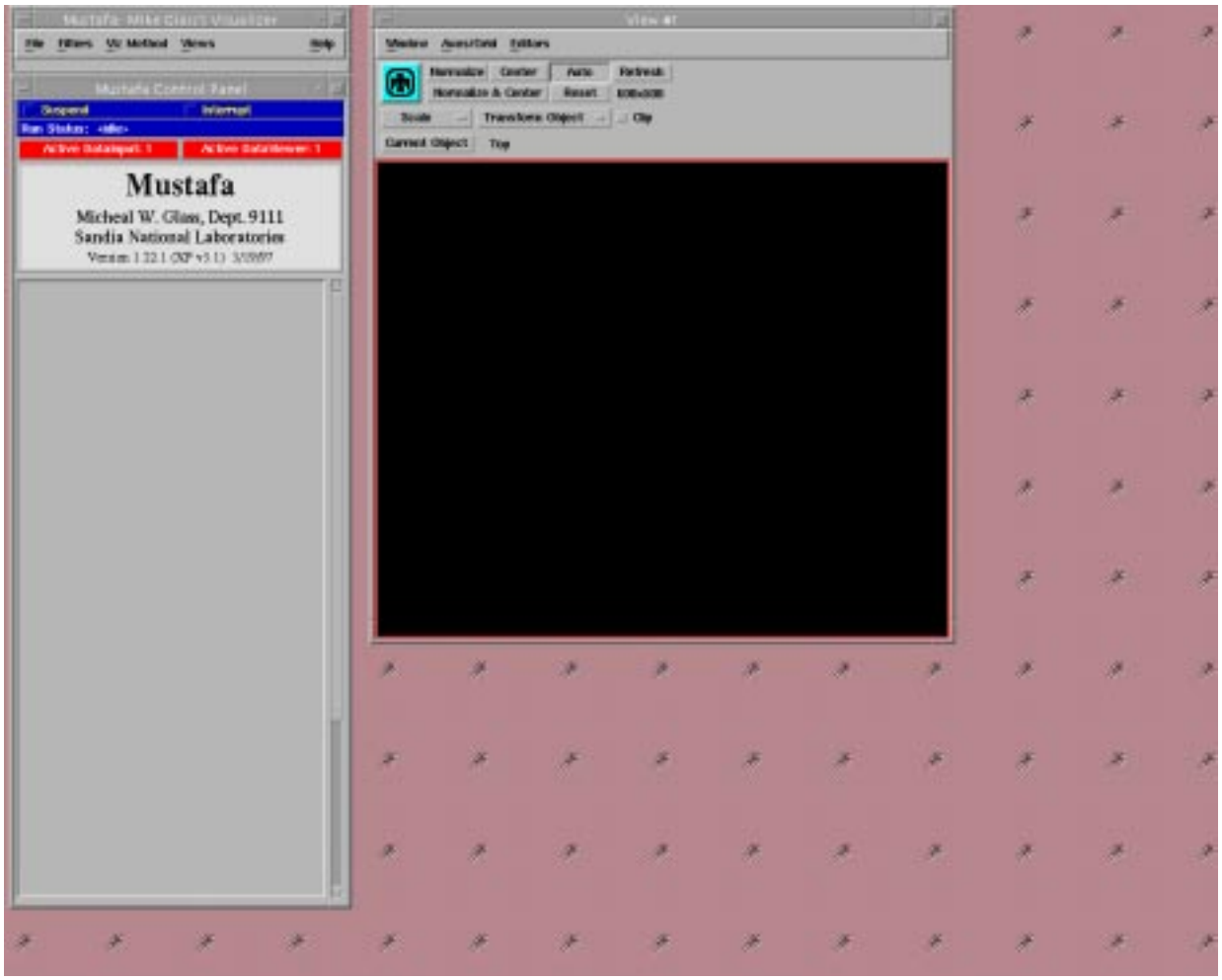


Figure 2.3 Startup screen.

2.3 Main Menu

Figure 2.4 Main menu bar

The menu bar is shown at the left provides access to the overall control of MUSTAFA. The *File* pulldown menu provides access to: (1) reading data files; (2) plotting of global variables; (3) ExodusII time animation; (4) data group creation, deletion, or selection; and (5) program exit. The *File→New Data Input* menu choice creates new data groups for reading in additional data files. The *File→Select Data Input* menu choice is used to select which data group is to be the active data group. The *File→Delete Data Input* menu choice deletes specific data groups. When a data group is deleted, all the visualization methods attached to it are also deleted. The *Filters* pulldown menu provides access to the global filters associated with the currently active data group. The *Viz Methods* pulldown menu provides access to all the visualization methods including selecting, deleting and turning on/off the individual methods. The *Viz Methods→Select* menu choice is used to select which viz method's GUI is to be made visible in the control panel. The *Viz Methods→Delete* menu choice deletes specific viz methods. The *Viz Methods→Visible* menu choice is used to control the visibility of the viz methods in the viewers without deleting the viz method altogether. The *Views* pulldown menu allows you to create new view windows, switch back and forth between them, delete them, resize them, and toggle "*bounding box*" mode. In "*bounding box*" mode, all mouse driven transformations will display a bounding box instead of the fully rendered object. This mode is particularly useful when using the software renderer or when working with large models. When a data viewer is deleted, all the visualization methods attached to it are also deleted. A few predefined window sizes can also be selected or the size of the viewer can be precisely set. Use the *Help* pulldown menu at your own risk.

2.4 Control Window

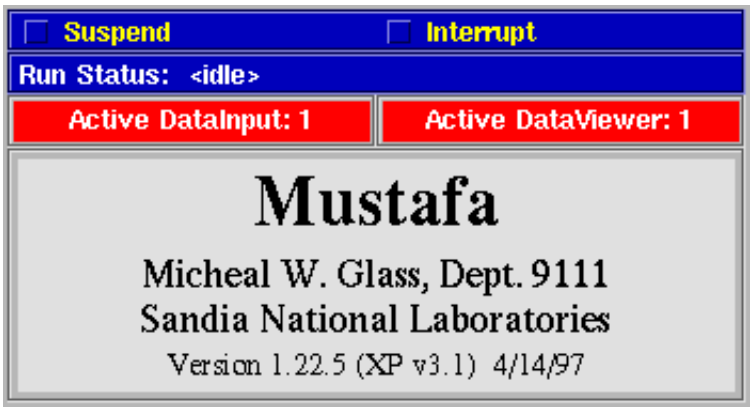


Figure 2.5 Status and control panel

The top portion of the control window is shown to the right. The *Suspend* toggle is used to temporarily suspend execution of the Express Object Manager (OM). When working with large models, it's useful to suspend the OM to change a bunch of GUI settings then resuming operation just before the last GUI setting is changed. The *interrupt* toggle will send an interrupt signal to interruptible actions (such as viewer update). The *Run Status* line displays the current activity that MUSTAFA is performing. The two red colored panels indicate the currently active data group and viewer. That is to say, newly instanced visualization methods will receive their data input from the currently active data group and send their output to the currently active viewer. When visualization methods are selected, their control panels will be viewable in the larger scroll window. At the top of the scroll window will be a red

button with the visualization method's title, number, data group number for that method, and the viewer number for that method. The format of the title will be:

G<data group number>:V<viewer number>: <visualization method title> #<method number>

Clicking on this title button will also popup a *Local Filters* window providing access to the local filters associated with that visualization method. If a particular visualization method (e.g. annotations) does not any local filters associated with it, clicking on the red title button will have no affect. The above title format is also used in the *Select*, *Delete*, and *Visible* cascaded menus in the *Viz Method* pulldown menu of the main menubar.

2.5 The HSV Color Model

MUSTAFA uses the HSV (Hue, Saturation, and Value) color model to specify colors. This color model is based on an artist's conception of tint, shade, and tone. The HSV color model is considered to be more intuitive than the RGB color model. The user selects a color (tint) by selecting a Hue and then adds or subtracts white or black. White is added/subtracted (shade) by adjusting the Saturation and black is added/subtracted (tone) by adjusting the Value. The range of all the parameter is 0.0 to 1.0. The model is shown in Figure 2.6 as a cone. Black is at (0.0, 0.0, 0.0) and white is at (0.0, 0.0, 1.0).

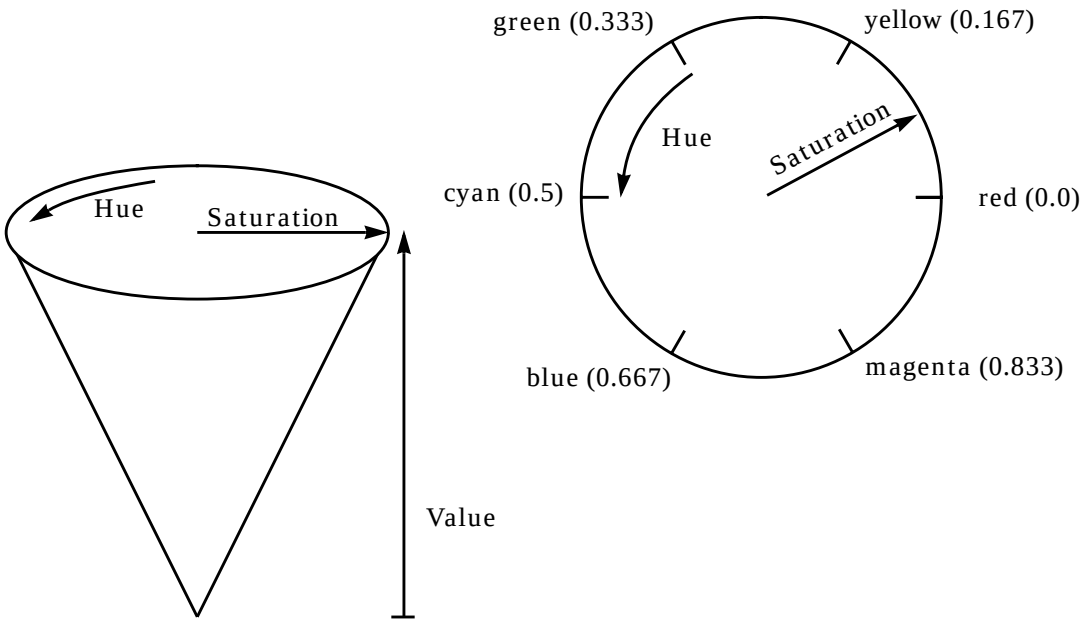


Figure 2.6 HSV color model

3. ExodusII Reader and Related Components

3.1 Exodus II Reader

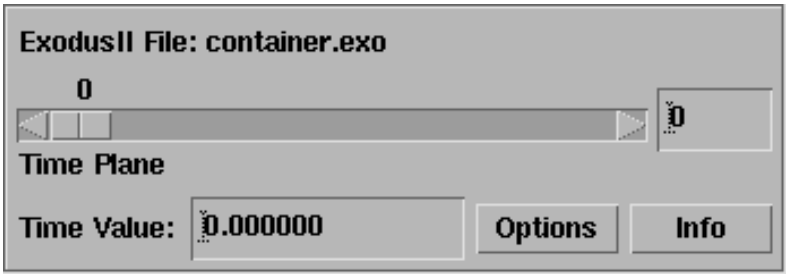


Figure 3.1 Exodus II Control Panel

When an EXODUS II file has been selected for reading (from the *File* pulldown menu of the main menubar), the top of the main control window will change to show the EXODUS II GUI as shown at the right. The current file name is displayed along with controls to access the various data slices present in the database. The time plane can be accessed with via the slider widget or directly typed into the typein widget next to the slider. Or, a specific time value can also be typed into the *Time Value* typein widget. If a time value is selected that is not precisely represented in the database, linear interpolation is used to determine the data values at the selected time value. The Info button. The *Options* button will display a popup window which can be used to specify which nodal and element data components are read from the database and available to the visualizer. When a new file is selected, all the nodal and element data is read. After this first pass, the *Options* popup window can be used to pare down the amount of data that is read from disk for each subsequent time selection. It also controls which of the selected element data components are interpolated to node data and which of the element data components are used to build a multi-material map as an additional data component. As mentioned earlier, the two red colored panels at the top indicate the currently active data group and viewer.

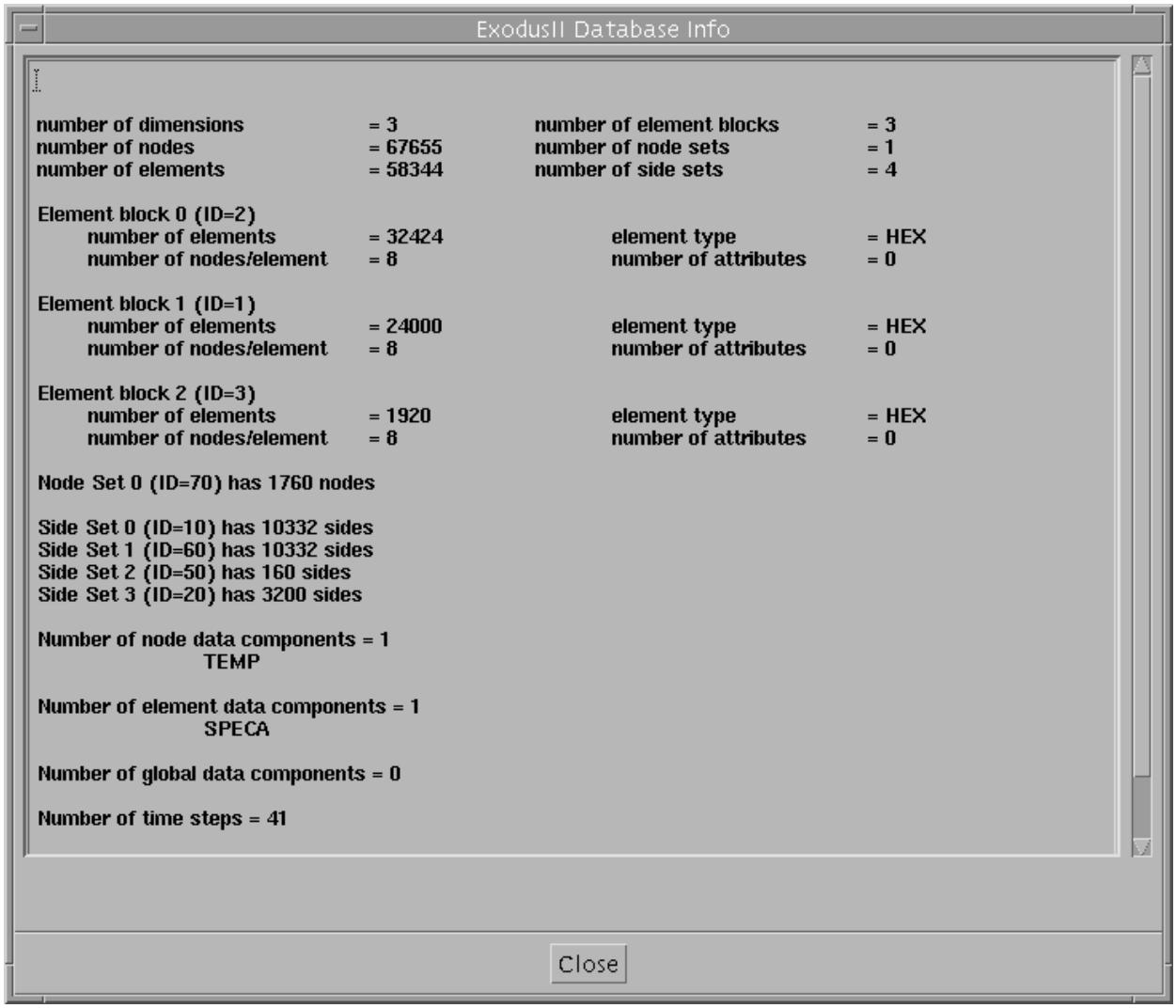


Figure 3.2 ExodusII file information window

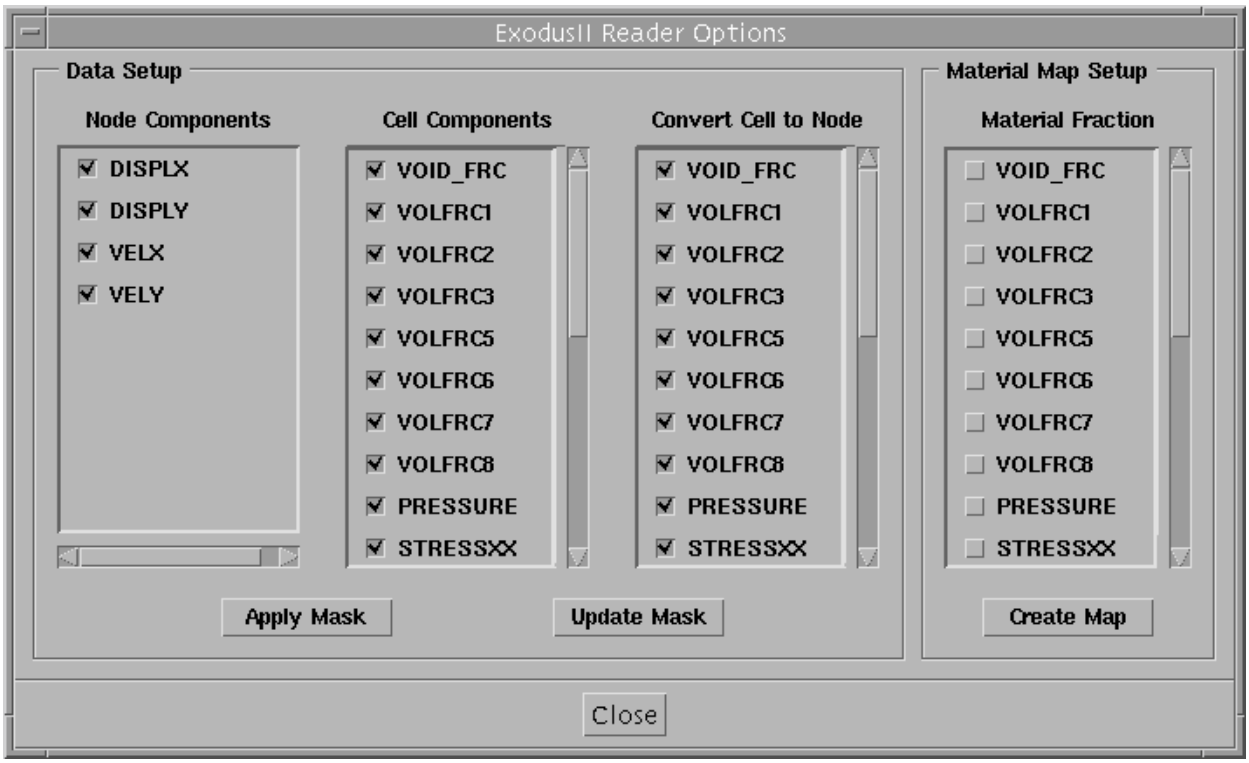


Figure 3.3 Variable load control panel

3.2 History Plots

Selecting the *File→History Plots* option will

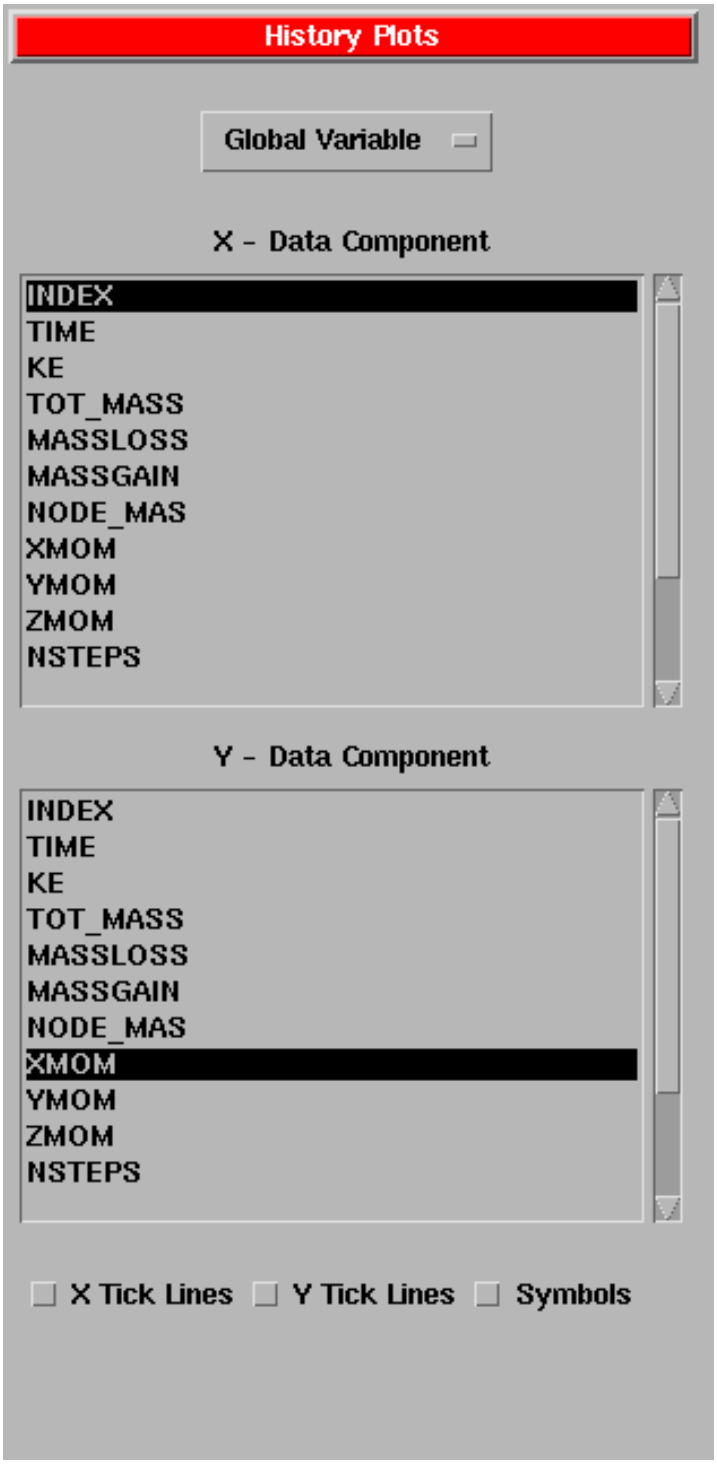


Figure 3.4 Control panel for "Global Variable History Plot"

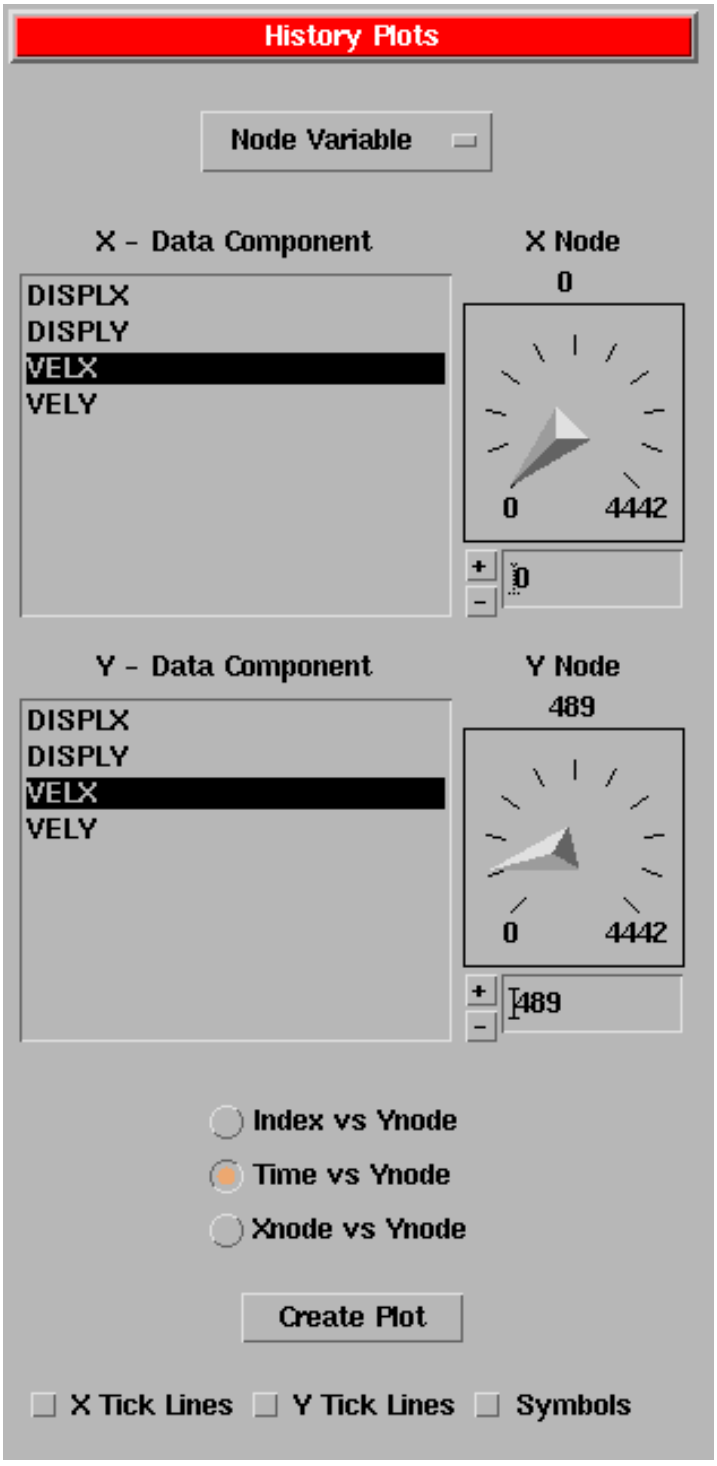


Figure 3.5 Control panel for "Nodal Variable History Plot"

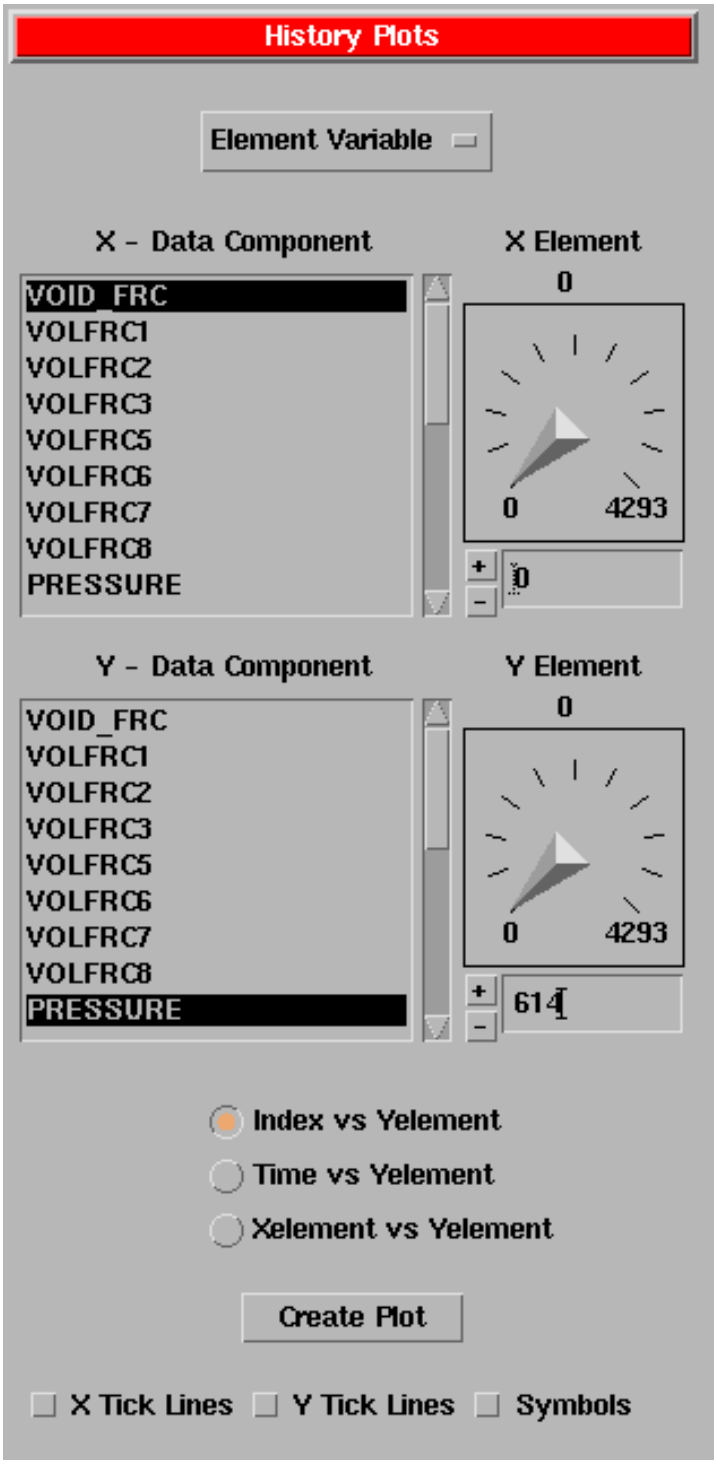


Figure 3.6 Control panel for "Element Variable History Plot"

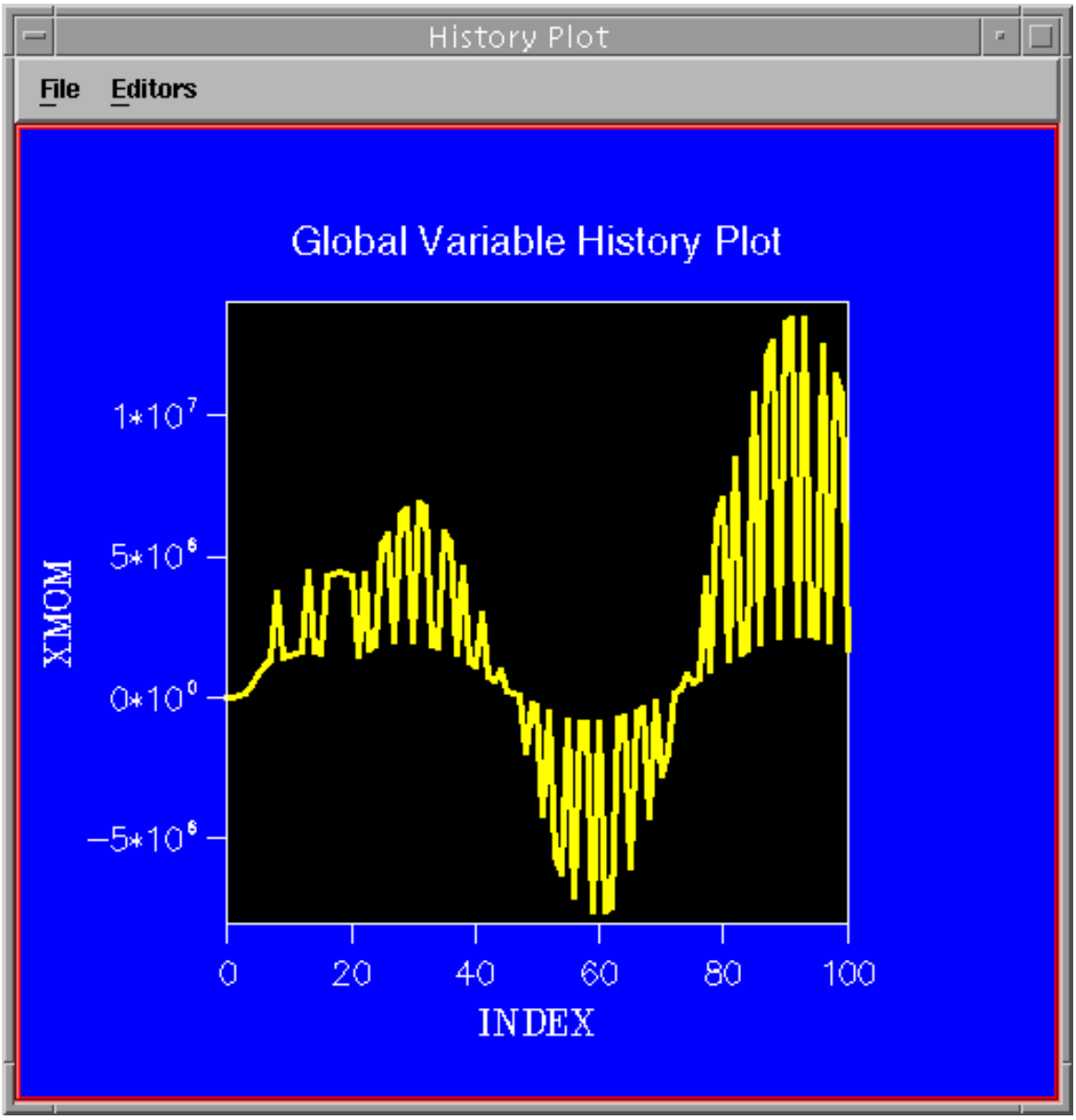


Figure 3.7 Sample global history variable plot.

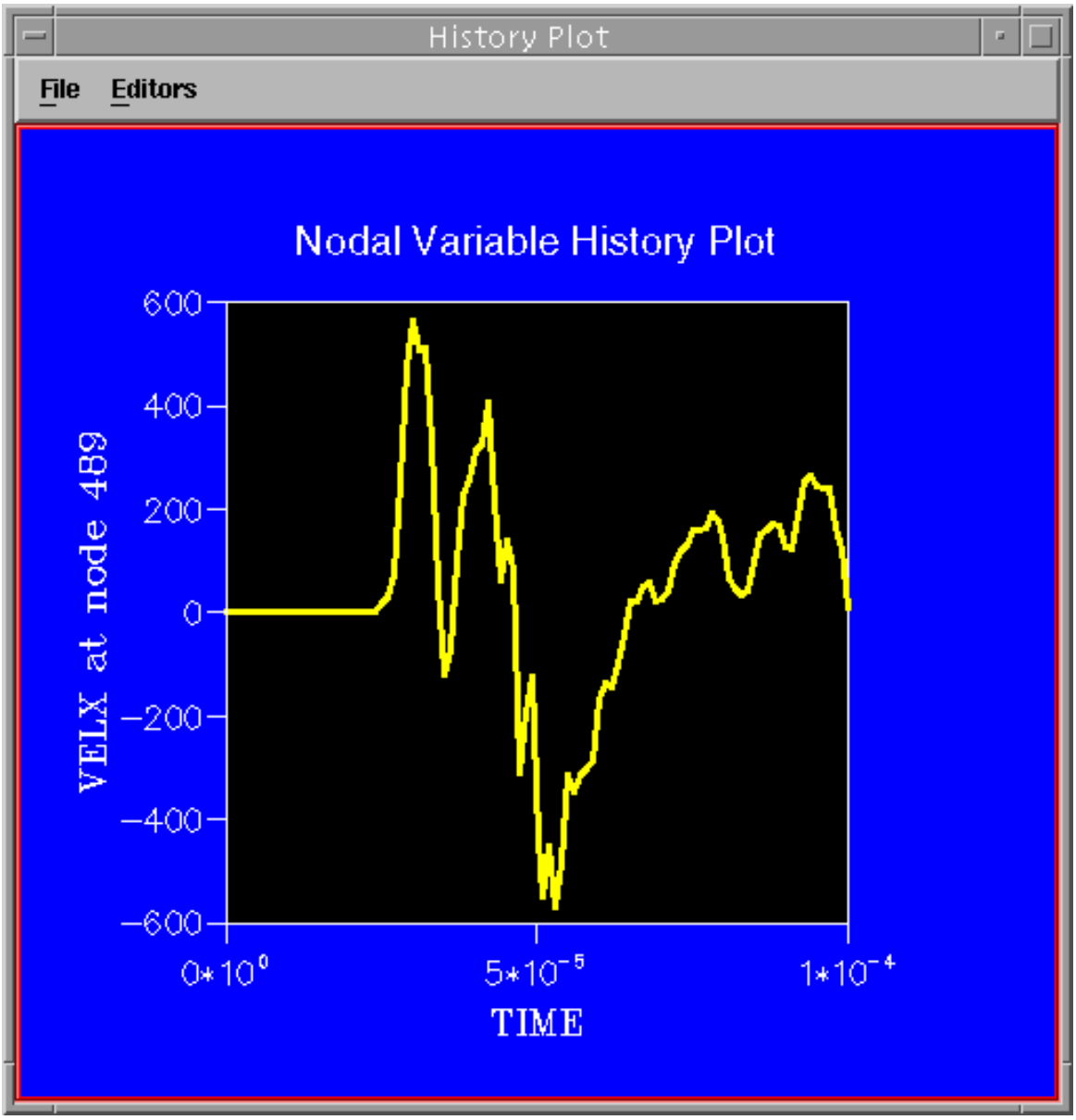


Figure 3.8 Sample nodal history variable plot.

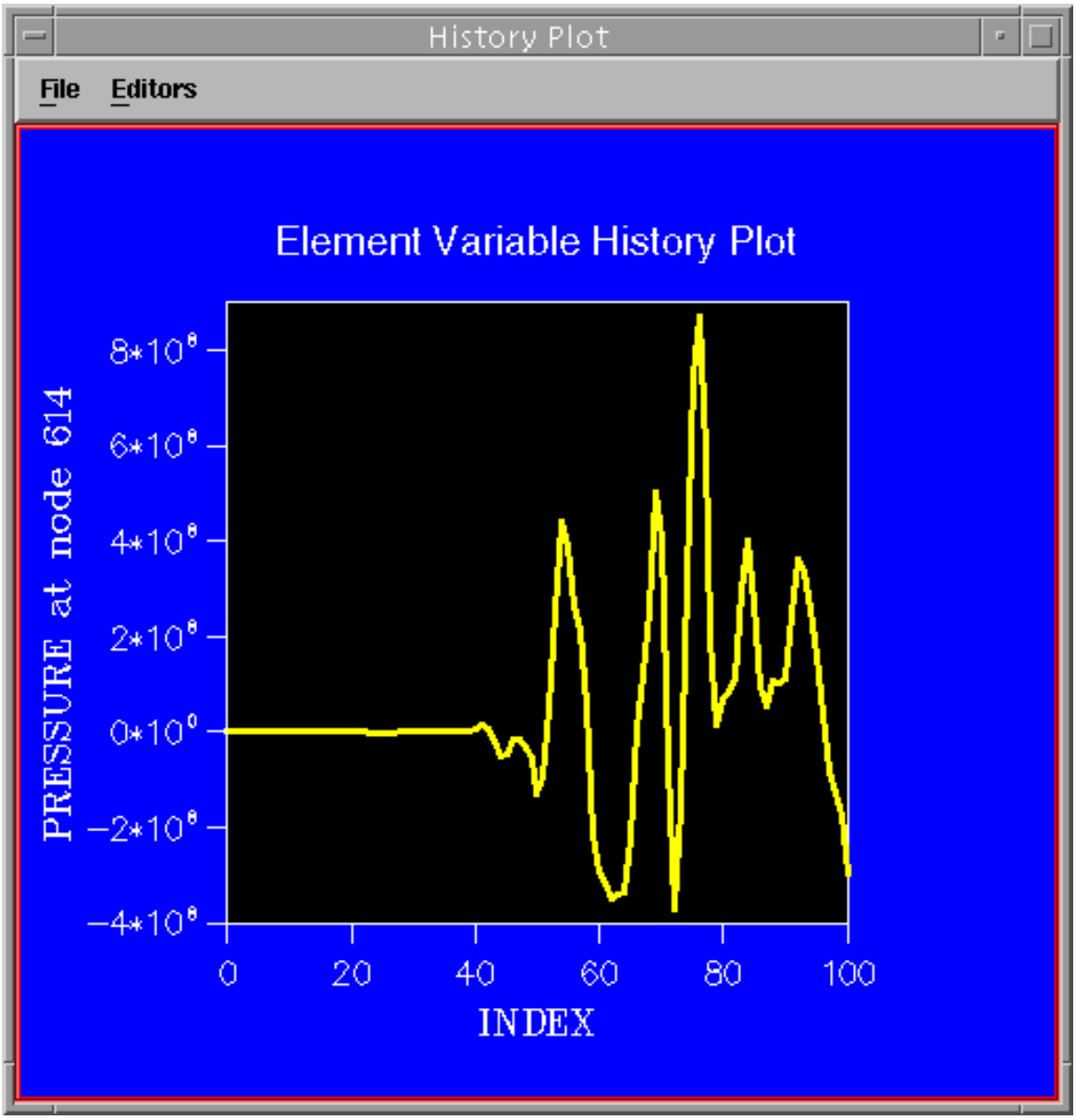


Figure 3.9 Sample Element variable history plot.

3.3 Time Animate

This module is used to automatically step through an ExodusII database containing transient data. The *Start Time* slider and typein widgets are used to specify the start time of the animation. The *End Time* slider and typein widgets are used to specify the end time of the animation. The Step slider and typein widgets are used to specify the time step used for advancing the animation. The *Run* toggle widget begins the animation sequence. The *Reset TIme* toggle widget resets the time to the beginning of the animation. The *Cycle* toggle widget puts the animation sequence in a continuous loop.

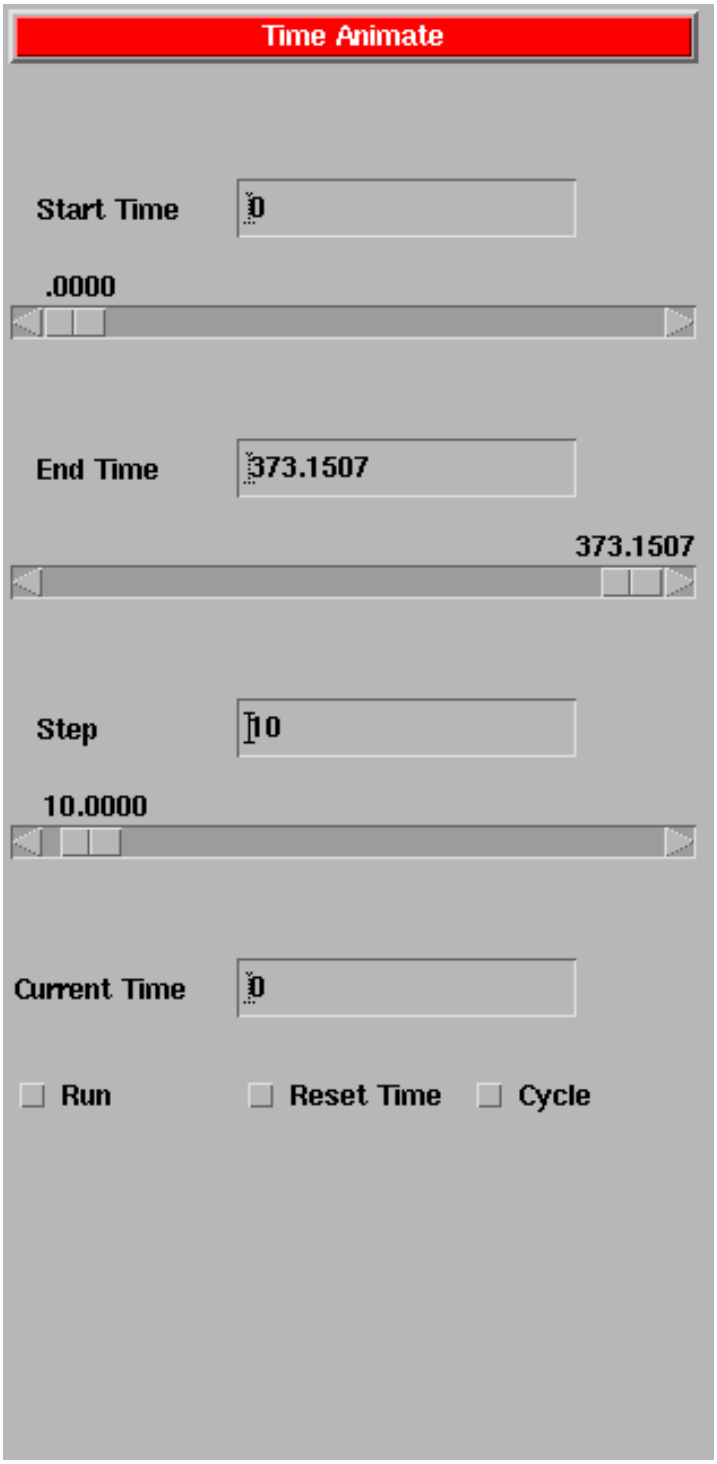


Figure 3.10 Control panel for “Time Animate”

3.4 Time Step Animate

This module is used to automatically step through an ExodusII database containing transient data. The *Start* slider and typein widgets are used to specify the start timestep of the animation. The *End* slider and typein widgets are used to specify the end timestep of the animation. The Step slider and typein widgets are used to specify the number of timesteps used for advancing the animation. The *Run* toggle widget begins the animation sequence. The *Reset* toggle widget resets the timestep to the beginning of the animation. The *Cycle* toggle widget puts the animation sequence in a continuous loop.

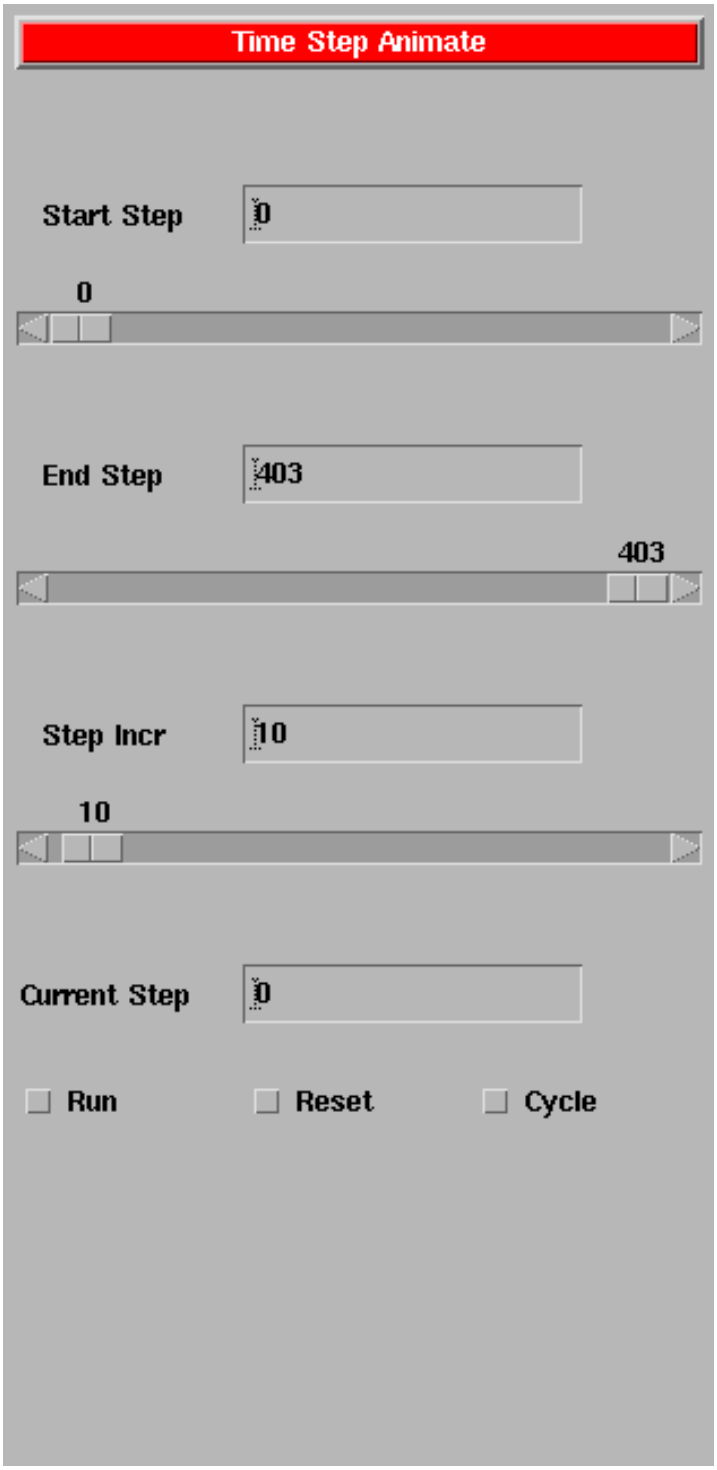


Figure 3.11 Control panel for “Time Step Animate”

4. Visualization Methods

1. Model
    1. Show Mesh/Geometry
        - Display the model's mesh/geometry.
    2. Show Element Blocks
        - Displays colored coded element blocks.
    3. Show Side Sets
        - Displays color coded side sets and surface normals.
    4. Show Node Sets
        - Displays color coded node sets with a colored glyph at each node in the node set.

2. Scalar
    1. External Faces
        - Displays the external surfaces color mapped to a specified variable.
    2. External Edges
        - Displays the external edges color mapped to a specified variable.
    3. Slice Plane
        - Displays the external edges color mapped to a specified variable.
    4. Iso Lines
        - Displays isolines of a given variable color mapped to a specified variable on either the external faces of the model or on a slice plane through the model..
    5. Iso Surfaces
        - Displays an isosurface of a given variable color mapped to a specified variable.
    6. Iso Volume
        - Displays isovolume of a given variable color mapped to a specified variable.
    7. Paint Cells:
        -
    8. MultiMaterial Map
        -
    9. Particle Fields
        -

3. Vector
    1. Hedge Hog: Displays arrow glyphs depicting a vector field
        -
    2. Streamlines
        -
    3. Particle Advection
        -

4. X-Y Plotting
    1. Node Variable vs. Distance
        -

5. Data Query
    1. Data Probe
        -

6. Annotations
    1. ExodusII Time Stamp
        -
    2. Global Variable Annotation
        -
    3. Nodal Variable Annotation
        -
    4. Element Variable Annotation
        -
    5. User Annotation
        -
    6. Sandia Logo
        -

7. Miscellaneous Editors
    1. Colormap Legend Editor
        -
    2. Line Attributes Editor
        -
    3. Line Probe Xform Editor
        -
    4. Plane Attributes Editor
        -
    5. Plane Probe Xform Editor
        -

4.1 Model Visualization Methods

4.1.1 Show Mesh/Geometry

This visualization method only shows the model and knows nothing about data components. Using the *Display Mode* popup menu, the mesh can be represented as just a wireframe model, or a wireframe model with hidden surface removal, or a solid object with mesh lines, or as a solid object without any mesh lines. In all cases, the color of the mesh lines and solid object can be independently set to any arbitrary color using an RGB color model and the individual slider widgets. The line width for the mesh lines can also be adjusted. When a solid model representation is requested, an additional control panel for the solid surface color is displayed.

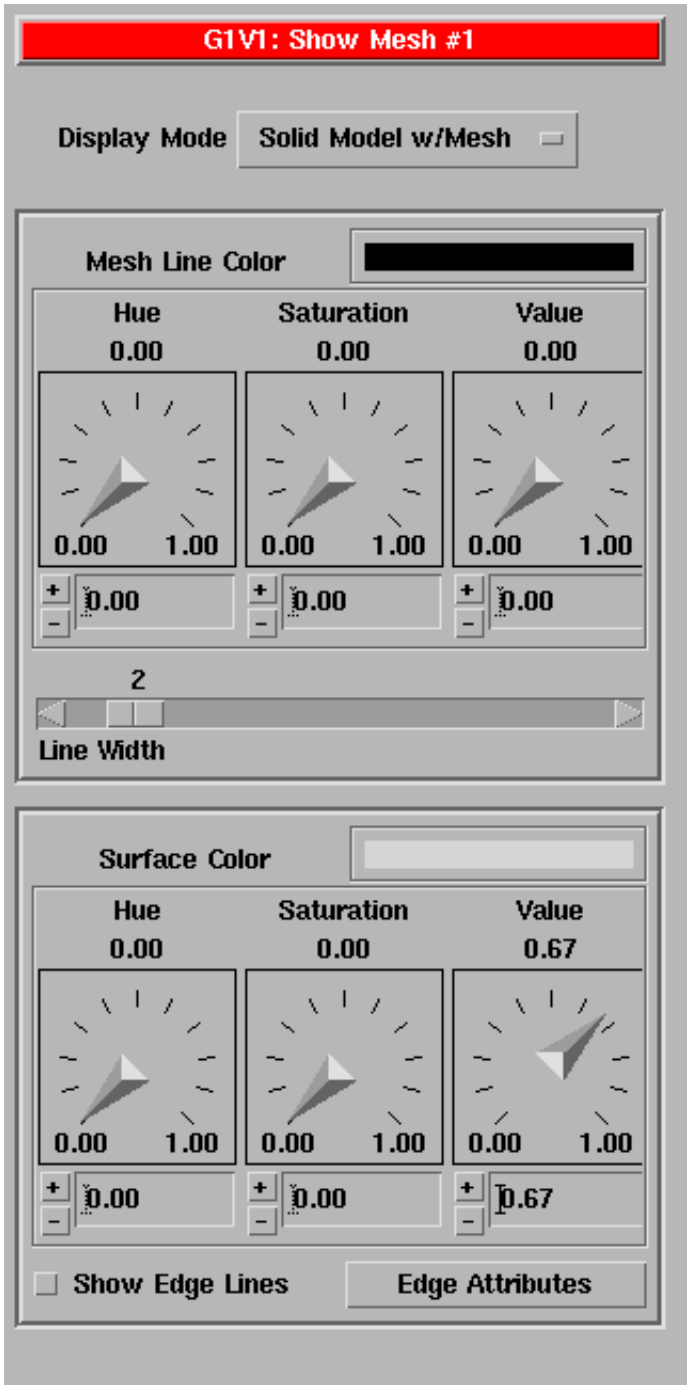


Figure 4.1 Control panel for “Show Mesh”

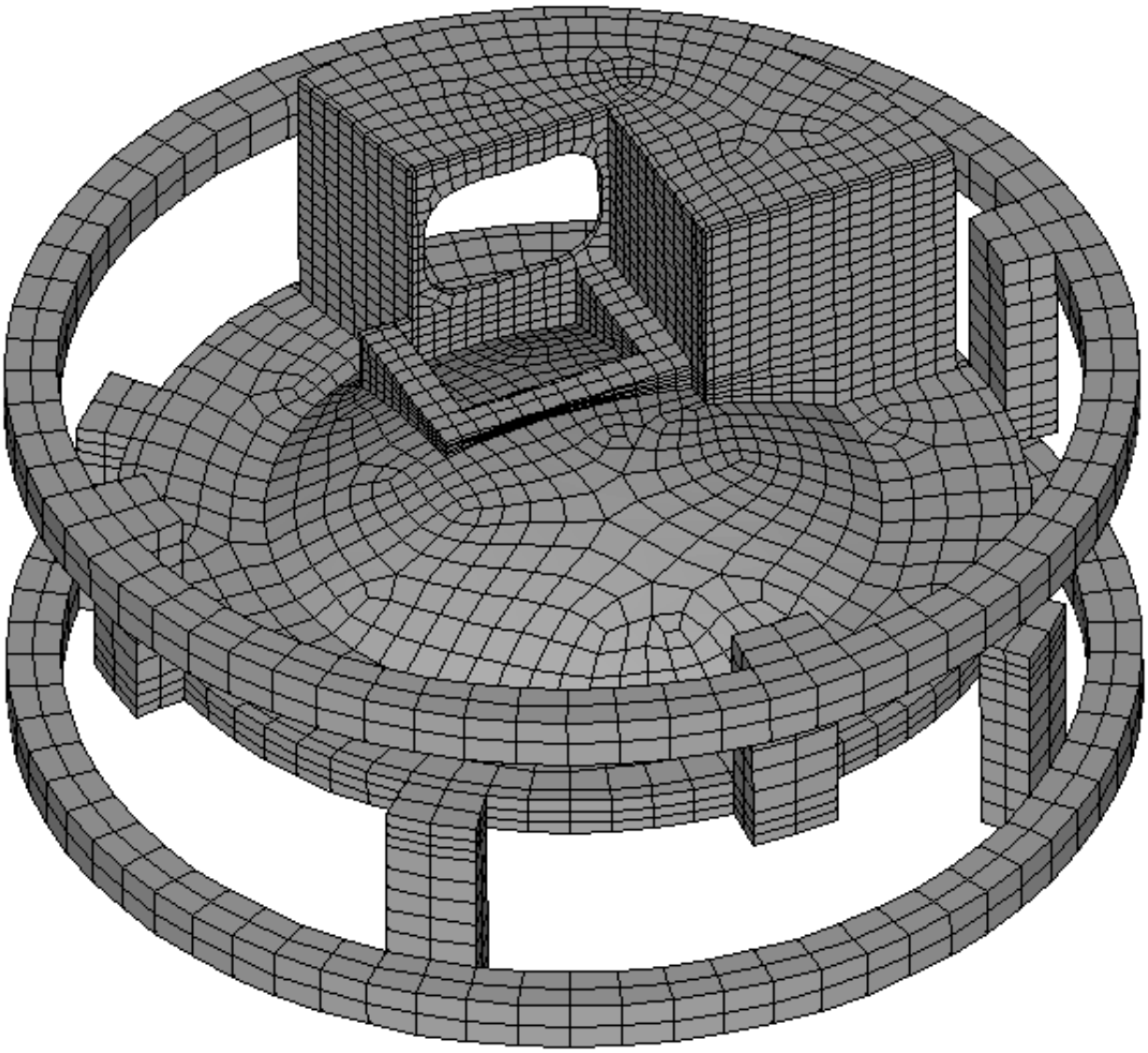


Figure 4.2 Sample “Show Mesh” viz method.

4.1.2 Show Element Blocks

This visualization method shows the element blocks that are present in the EXODUS II database. The individual element blocks are colored from blue to red based upon their order in the database. Each element block can be selectively turned on or off with their respective toggle widgets. Mesh lines on the element blocks can also be turned on with the *Show Mesh Lines* toggle. The mesh's line attributes (e.g. color and line width) can be adjusted by popping up a control panel with the *Line Attributes* button.

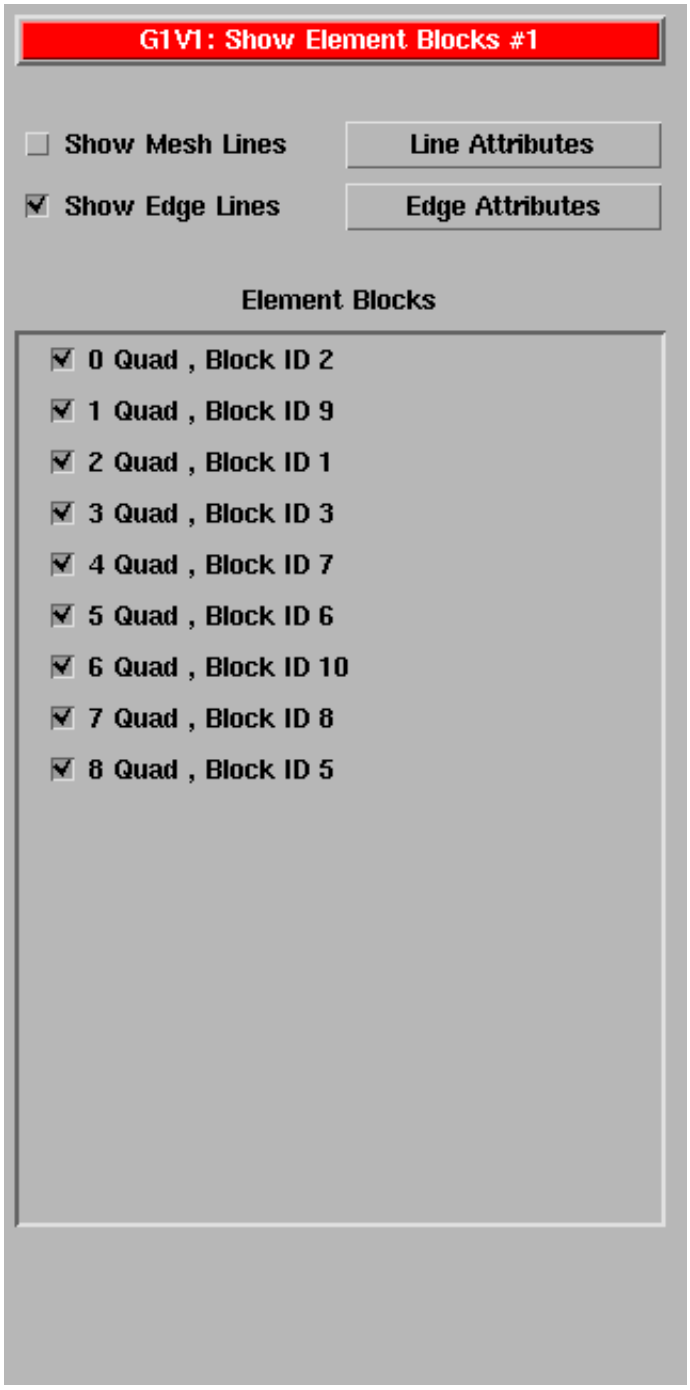


Figure 4.3 Control Panel for "Show Element Blocks"

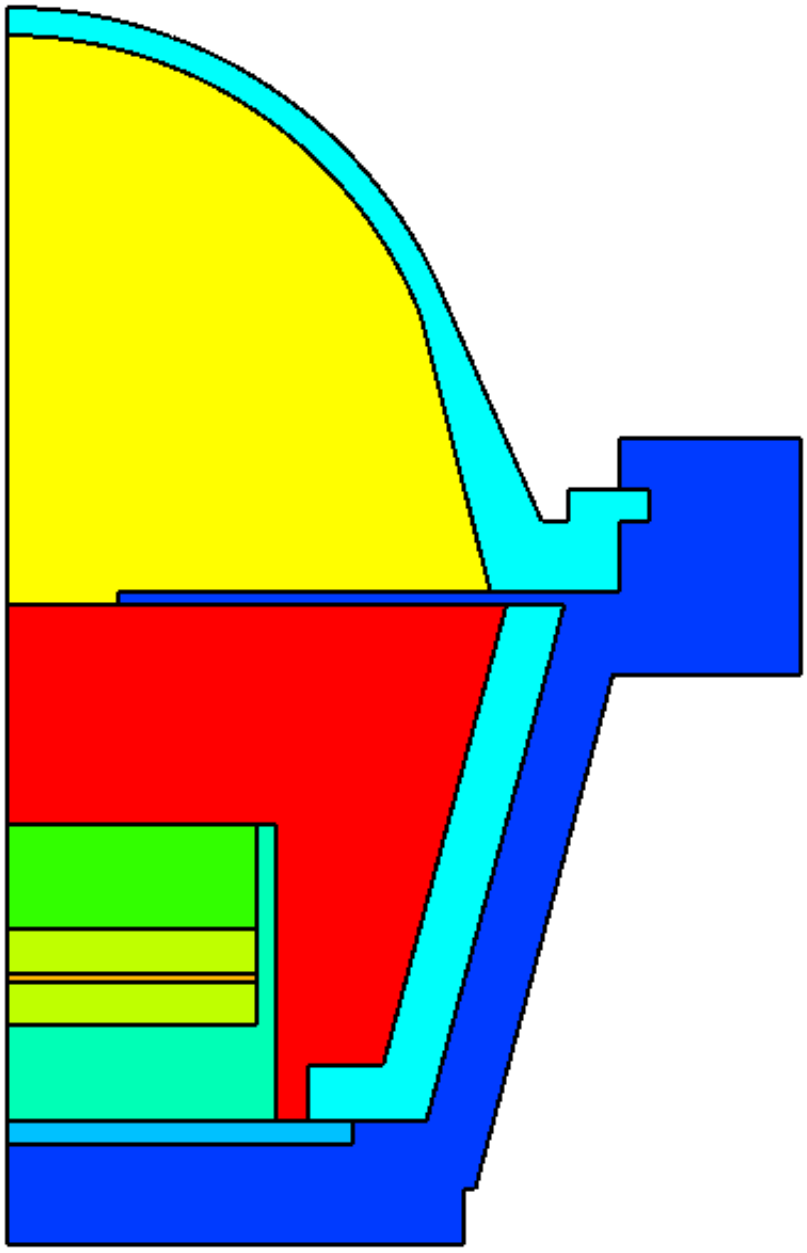


Figure 4.4 Sample “Show Elements” viz method.

4.1.3 Show Side Sets

This visualization method shows the side sets that are present in the EXODUS II database. The individual side sets are colored from blue to red based upon their order in the database. Each side set can be selectively turned on or off with their respective toggle widgets. Mesh lines on the side sets can also be turned on with the *Show Mesh Lines* toggle. The mesh's line attributes (e.g. color and line width) can be adjusted by popping up a control panel with the *Line Attributes* button.

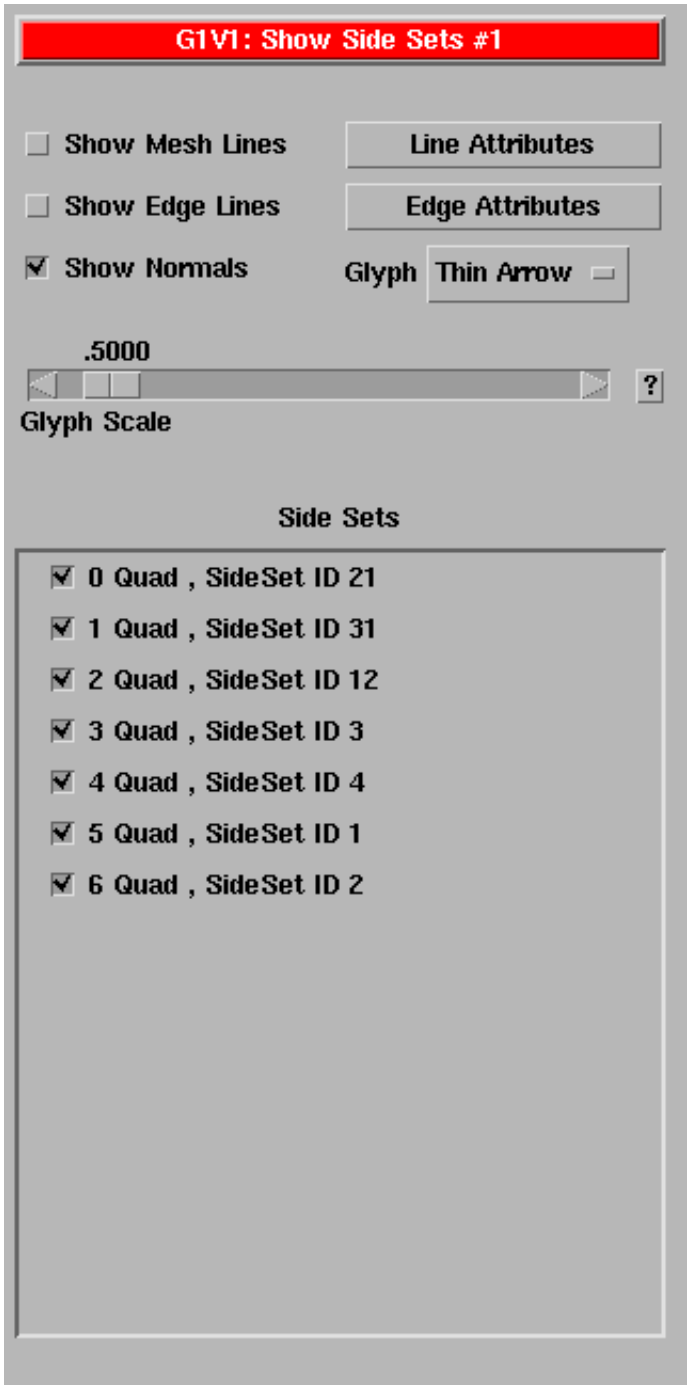


Figure 4.5 Control panel for "Show Side Sets"

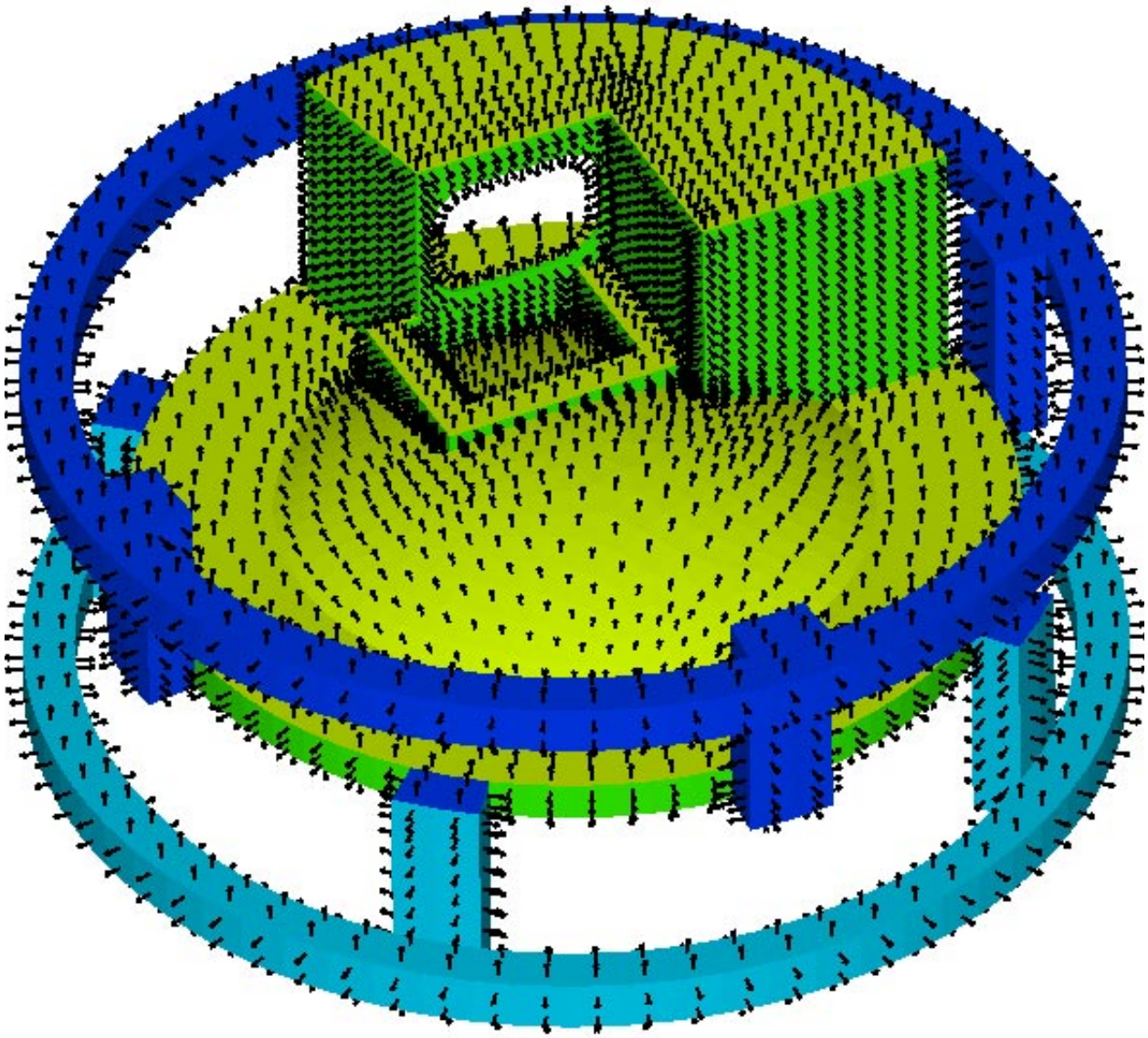


Figure 4.6 Sample “Show Side Sets” viz method.

4.1.4 Show Node Sets

This visualization method shows the node sets that are present in the EXODUS II database. The individual node sets are colored from blue to red based upon their order in the database. Each node set can be selectively turned on or off with their respective toggle widgets. The node position is marked with a colored glyph. The glyph can either be a point or a 3D diamond. If a diamond is specified, its size can be adjusted with the *Glyph Scale* slider.

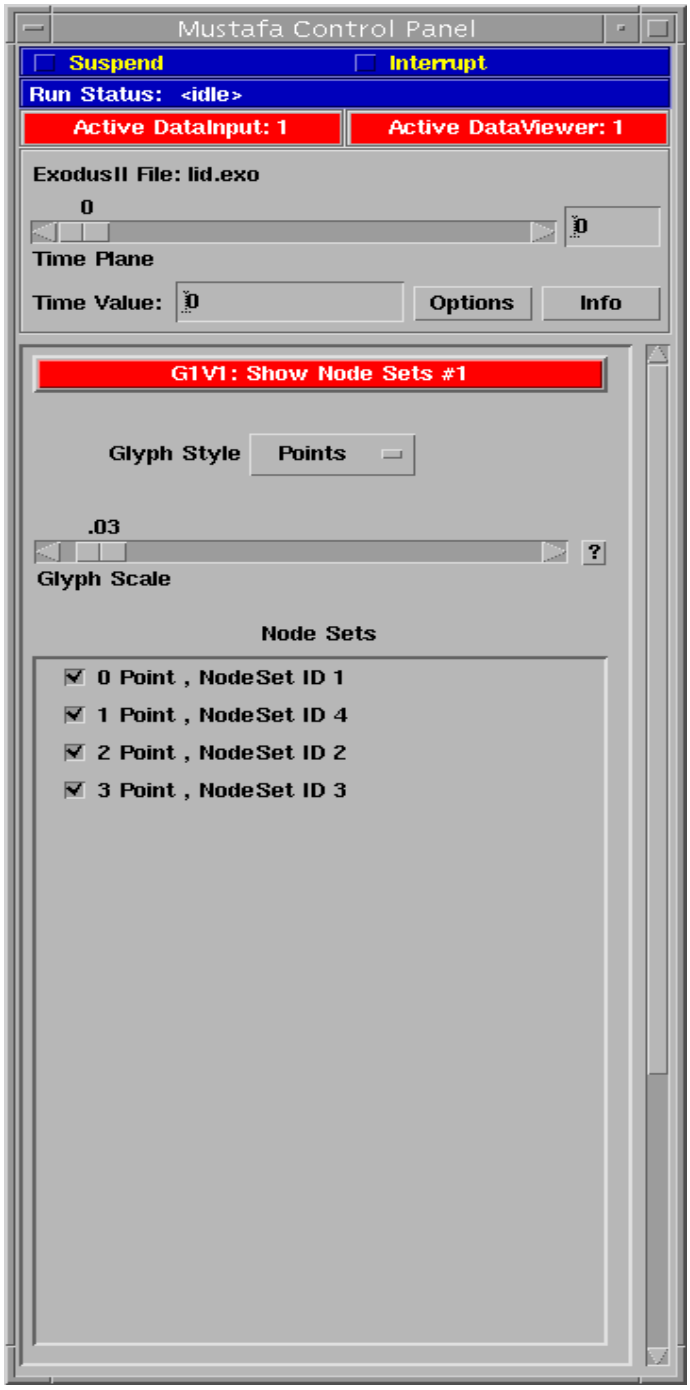


Figure 4.7 Control panel for “Show Node Sets”

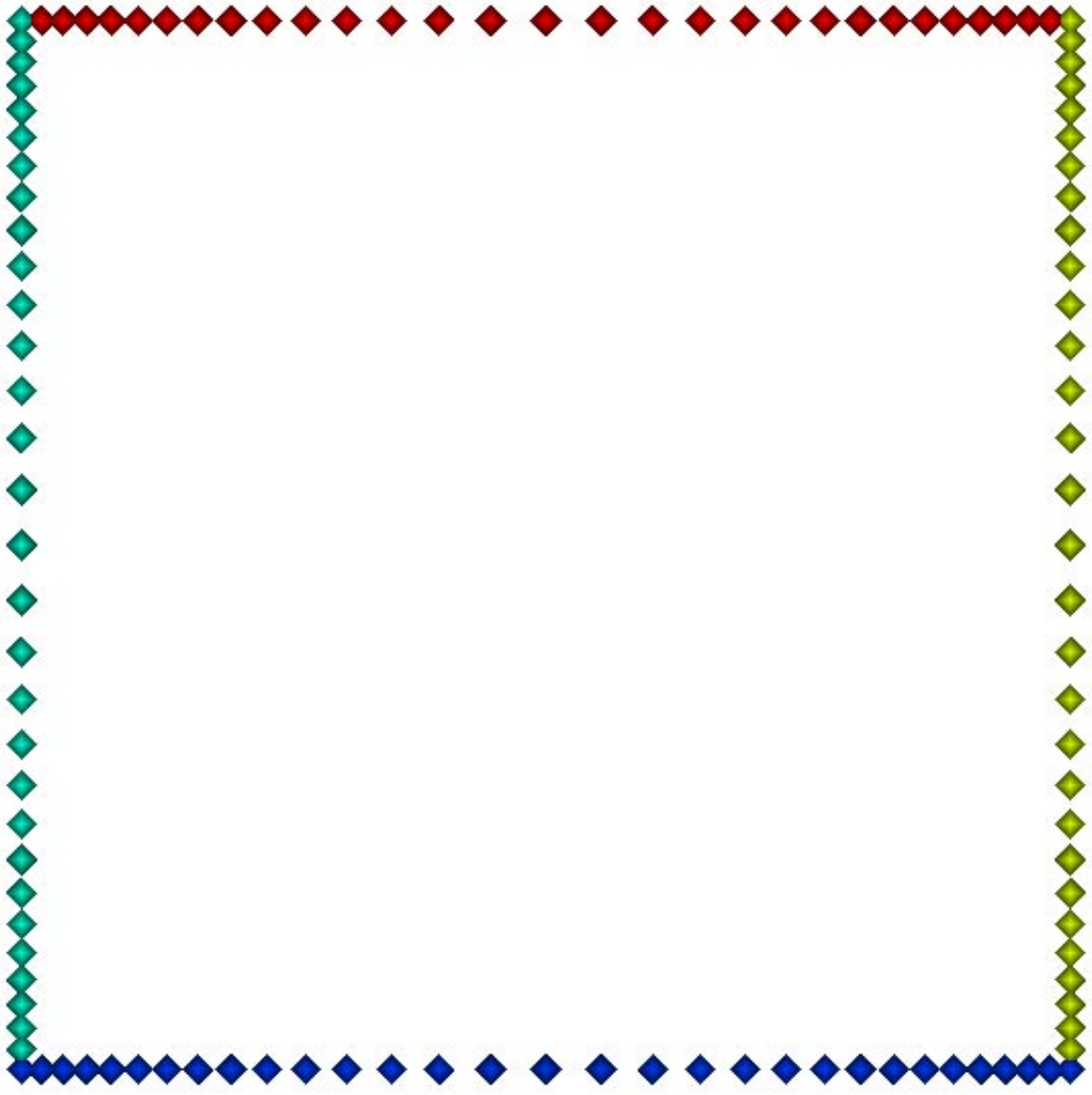


Figure 4.8 Sample “Show Node Sets” viz method

4.2 Scalar Visualization Methods

4.2.1 External Faces

This visualization method is for node based data. It constructs a solid fringe plot of the selected data component on the external faces of each element. Mesh lines and/or edge lines can also be displayed and the line attributes are also under user control. The *Data Map Editor* button will activate a popup window to control the data-to-color mapping and the *Data Map Legend* button will activate a popup window to display a legend in the view window. The *Legend On/Off* button will toggle the legend display on or off.

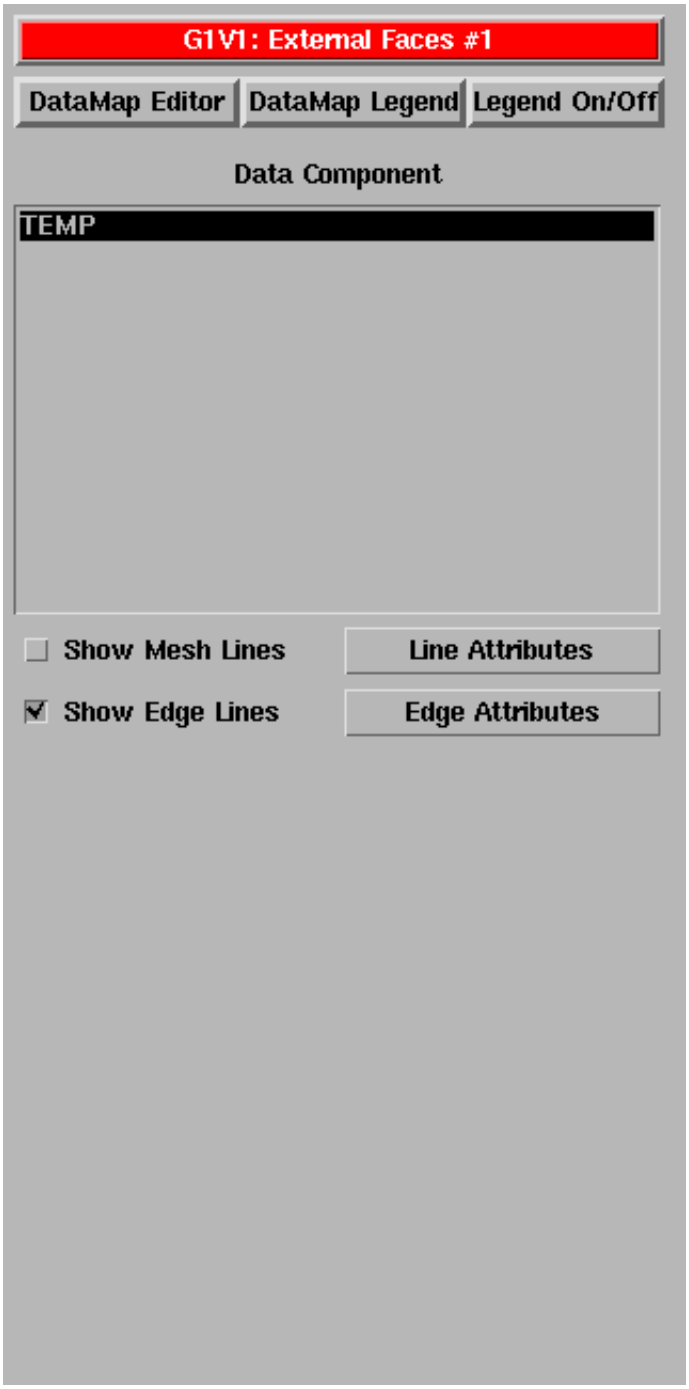


Figure 4.9 Control panel for “External Faces”

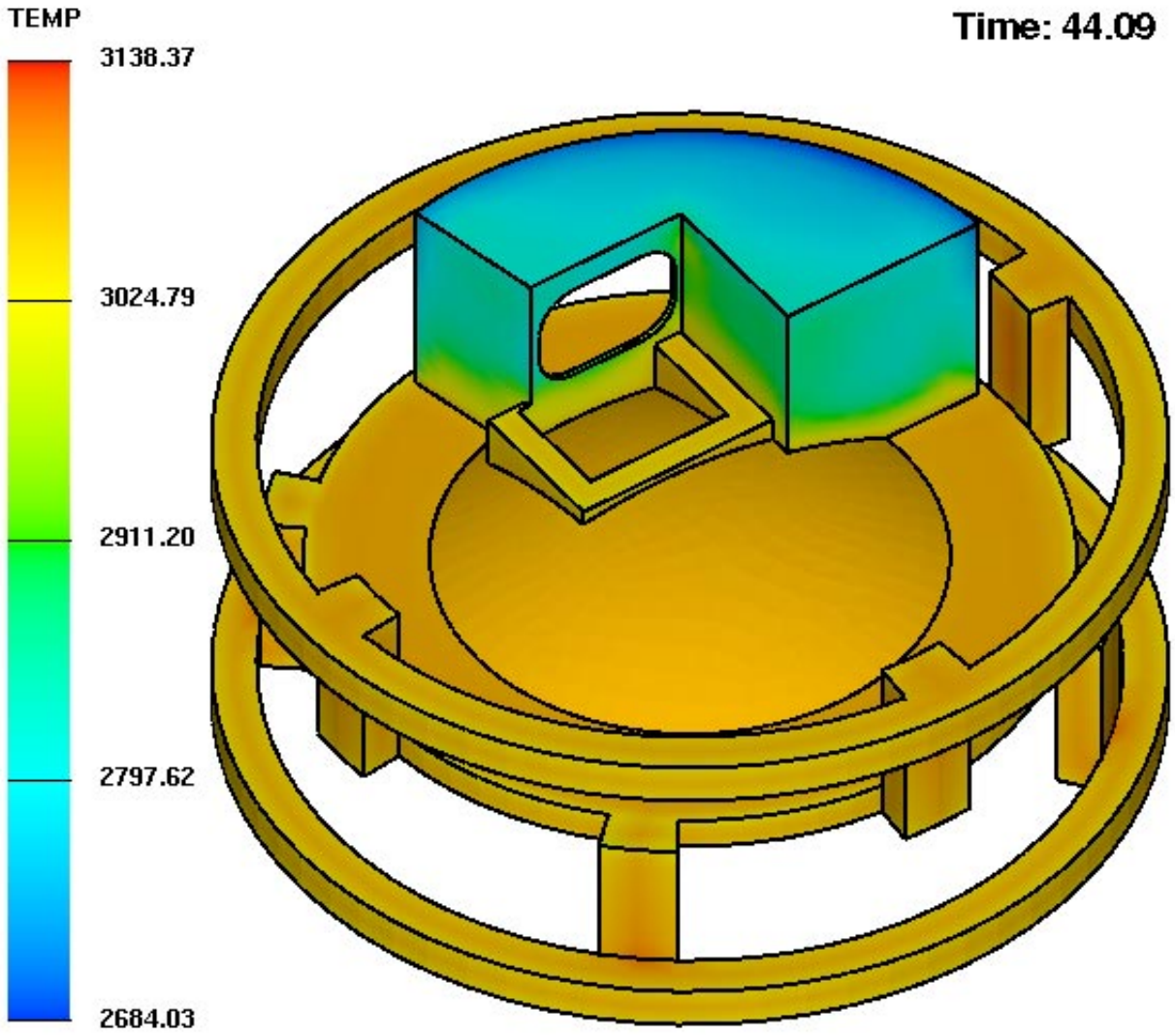


Figure 4.10 Example of “Exteranl Faces”

4.2.2 External Edges

This visualization method is for node based data. It constructs a wireframe representation of the model from the external edges. It is useful when you want to see objects produced from other visualization tools (e.g. iso surface, slice plane, etc.) while still being able to see the enclosing skeletal shape of the mesh. The edges can be colored according to a selected data component or can be colored according to some user selected constant value. The width of the lines can also be adjusted by the user. The *maximum edge angle* slider can be used to determine the accuracy of the boundary representation on the base of the angle between two adjoining faces. All edges that have an angle less than this value are represented in the output. The angle is given in degrees.

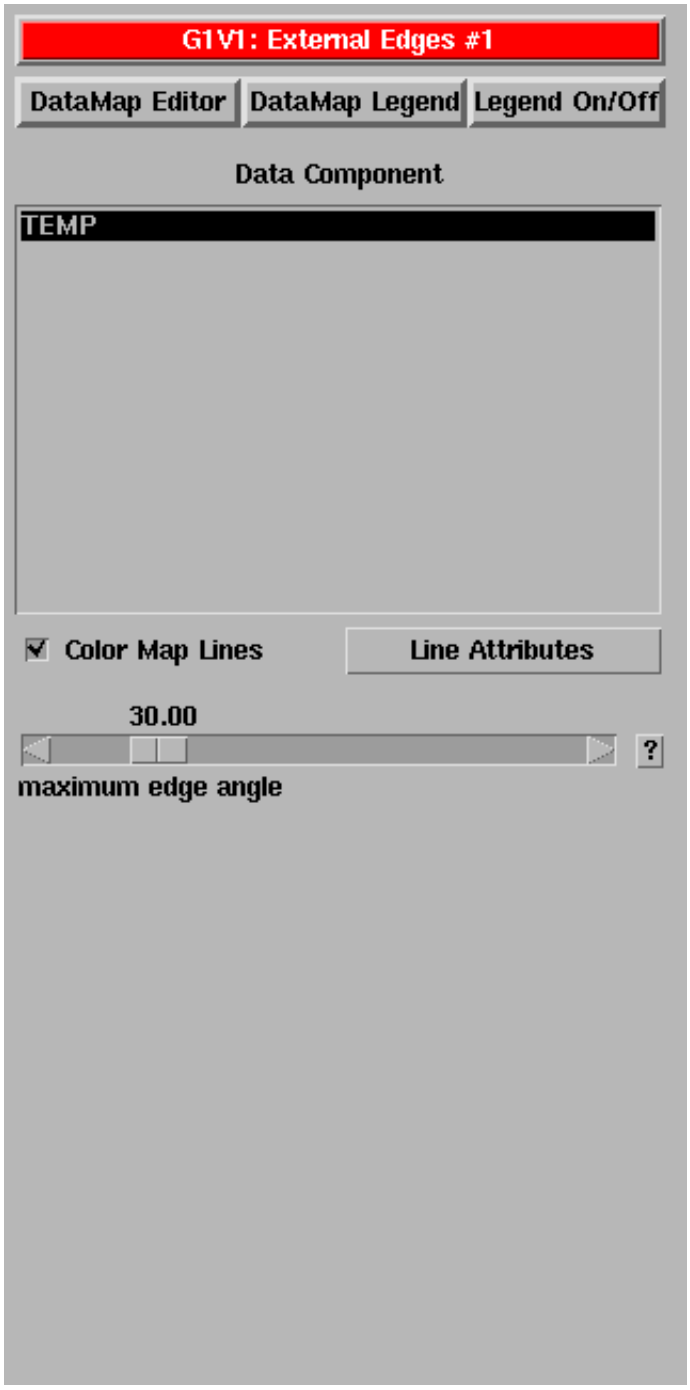


Figure 4.11 Control panel for “External Edges”

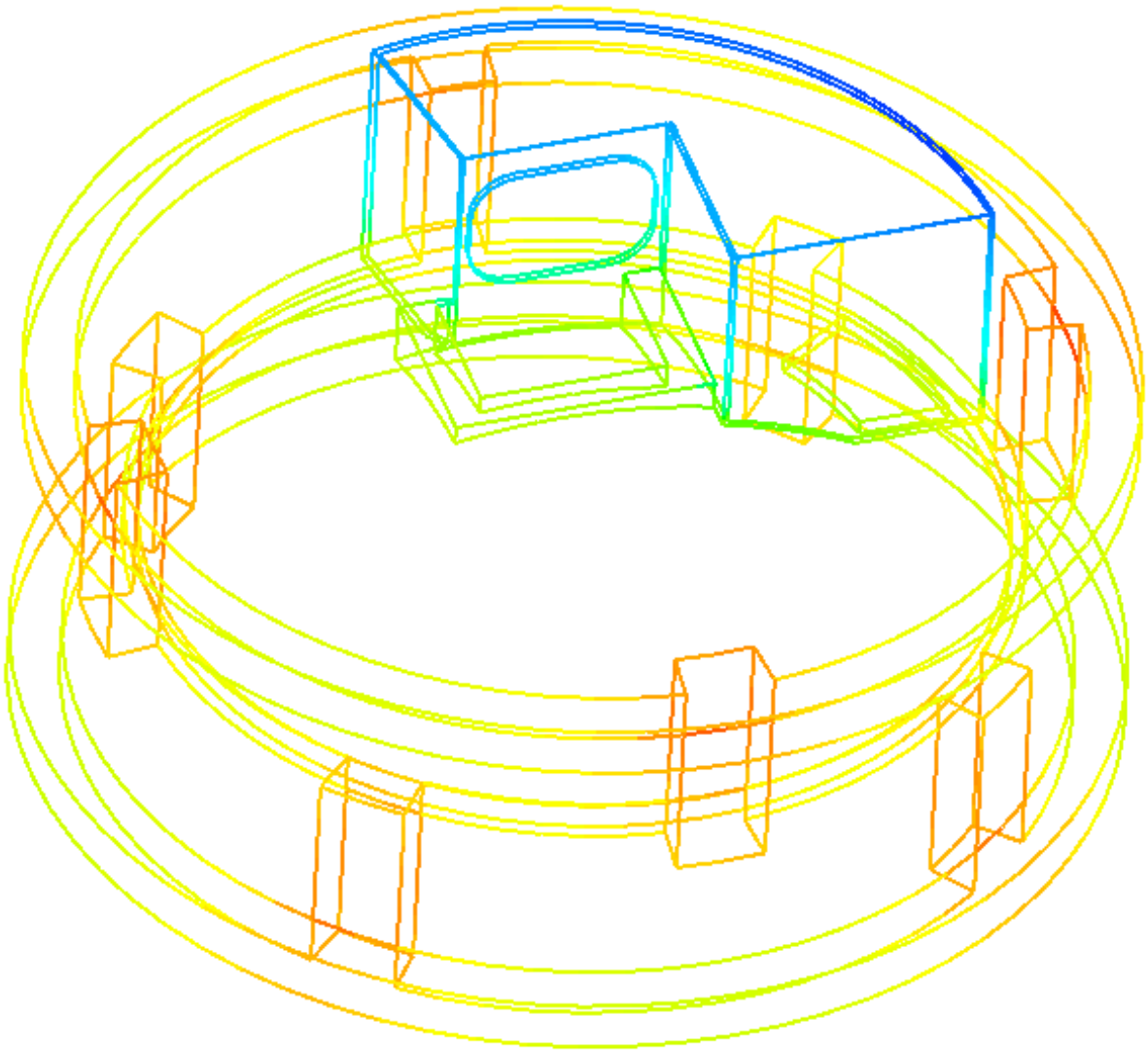


Figure 4.12 Sample of “External Edges” viz method

4.2.3 Slice Plane

This visualization method is for node based data. It passes a slice plane through the model and creates a fringe plot of the selected data component on the slice. The data component scroll window is used to select which data component is used for the fringe plot. The *plane distance* slider widget translates the slice plane along its normal direction. The *Probe Transform* button will activate a popup window to precisely control the geometric transformation of the slice plane. The *Plane Attributes* button will activate a popup window to set the attributes of the plane such as it's color and transparency. The *Data Map Editor* button will activate a popup window to control the data-to-color mapping and the *Data Map Legend* button will activate a popup window to display a legend in the view window. The *Legend On/Off* button will toggle the legend display on or off.

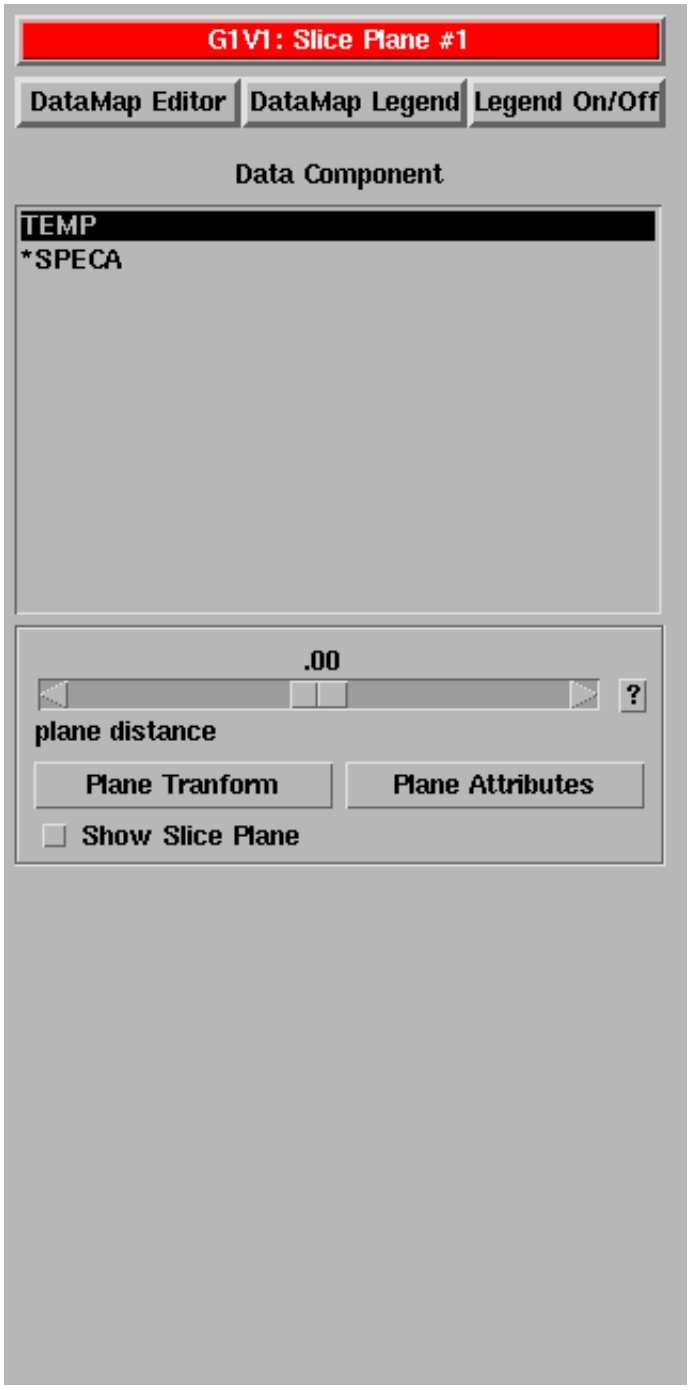


Figure 4.13 Control panel for "Slice Plane"

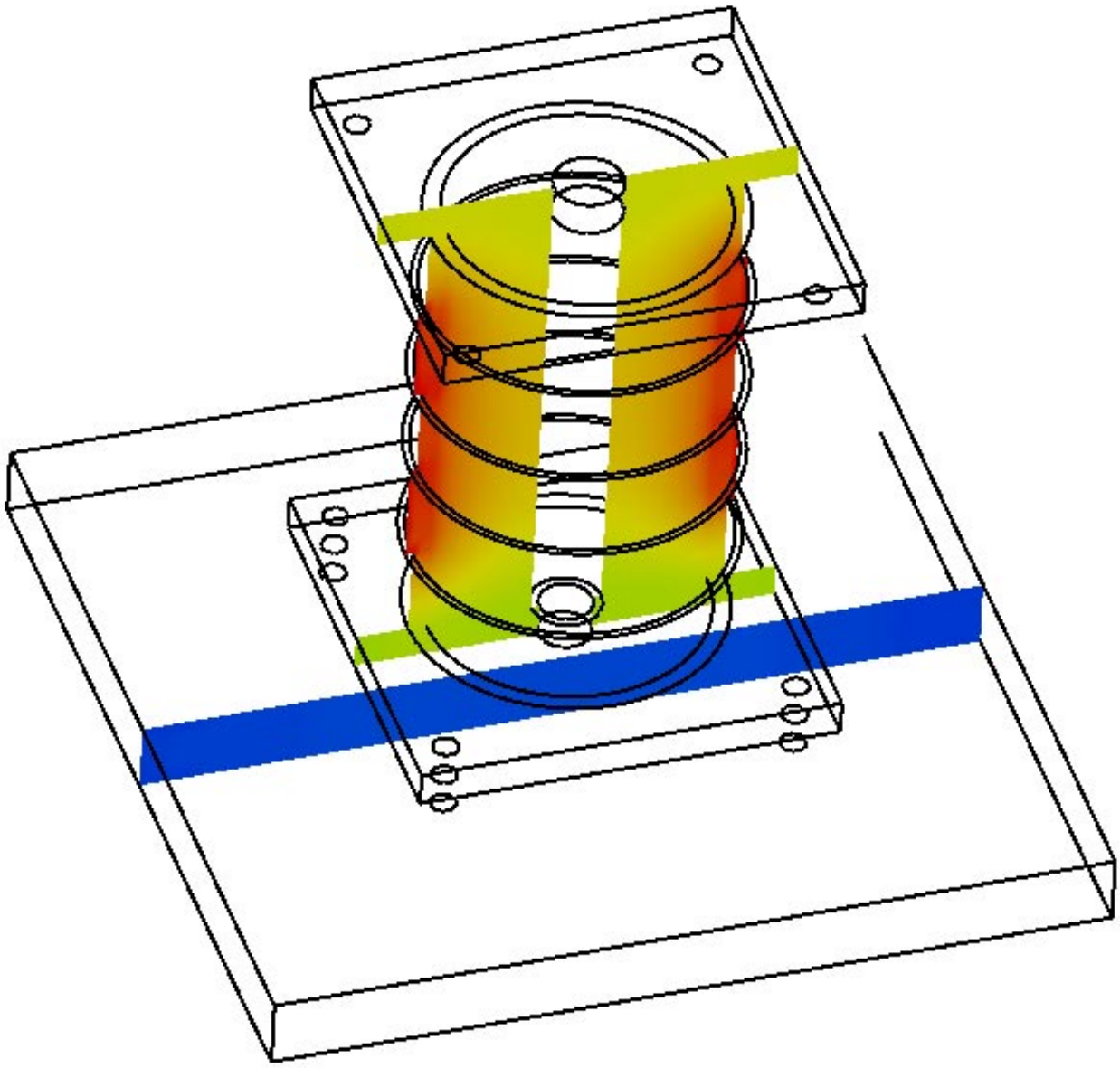


Figure 4.14 Example of “Slice Plane”

4.2.4 Iso Lines

This visualization method is for node based data. It draws lines of constant value. For 3-D data, the isolines can be drawn on either the external surface of the model or on a plane passing through the model. The data component scroll window is used to select which data component is used for the isolines. The number of contours (lines) is set by the *number of contours* slider widget. The *min level* and *max level* slider widgets select the lower and upper bounds of the selected data component between which the specified number of isolines will be drawn. The isolines may colored according to the data value by selecting the *Color Map Lines* toggle. If this toggle is not selected, the color of all the lines can be set by accessing the line's attributes via the *Line Attributes* button. The width of the lines can also be set via the *Line Attributes*. When isolines are to drawn on a plane passing through a 3-D dataset, additional controls are displayed to control the plane. The *Data Map Editor* button will activate a popup window to control the data-to-color mapping and the *Data Map Legend* button will activate a popup window to display a legend in the view window. The *Legend On/Off* button will toggle the legend display on or off.

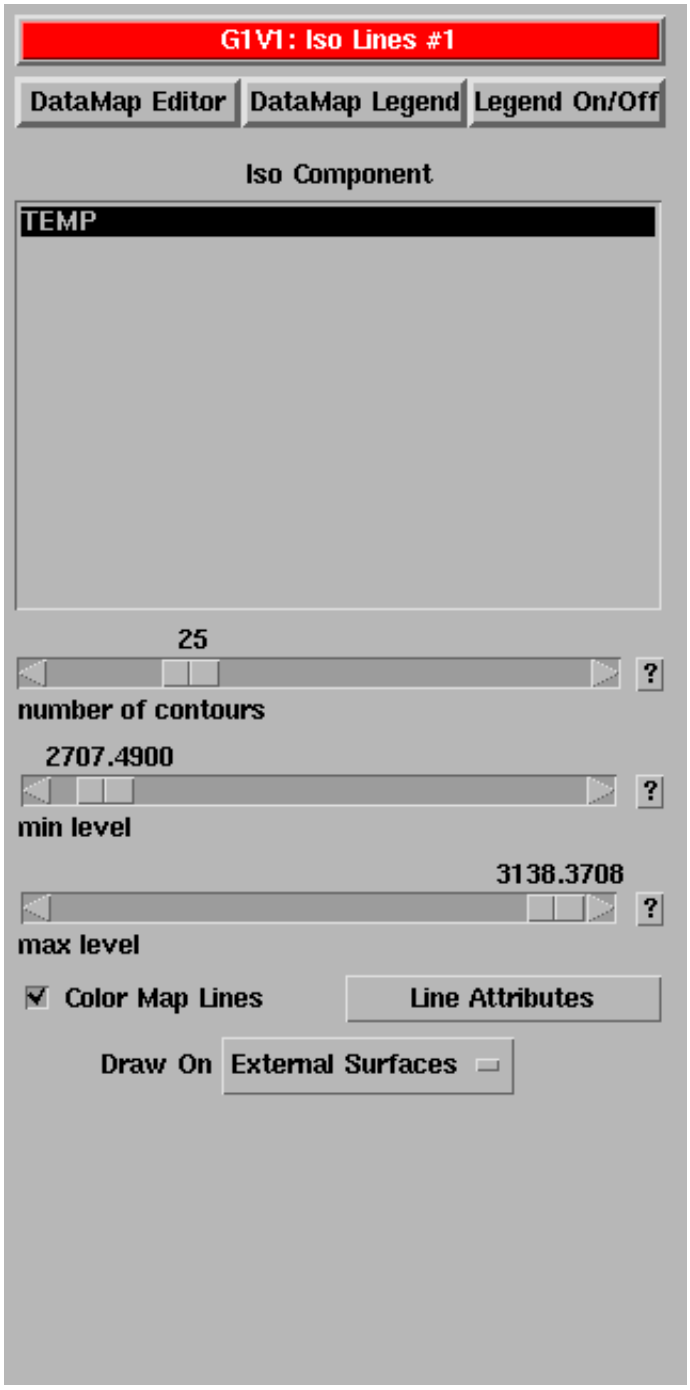


Figure 4.15 Control panel for "Iso Lines"

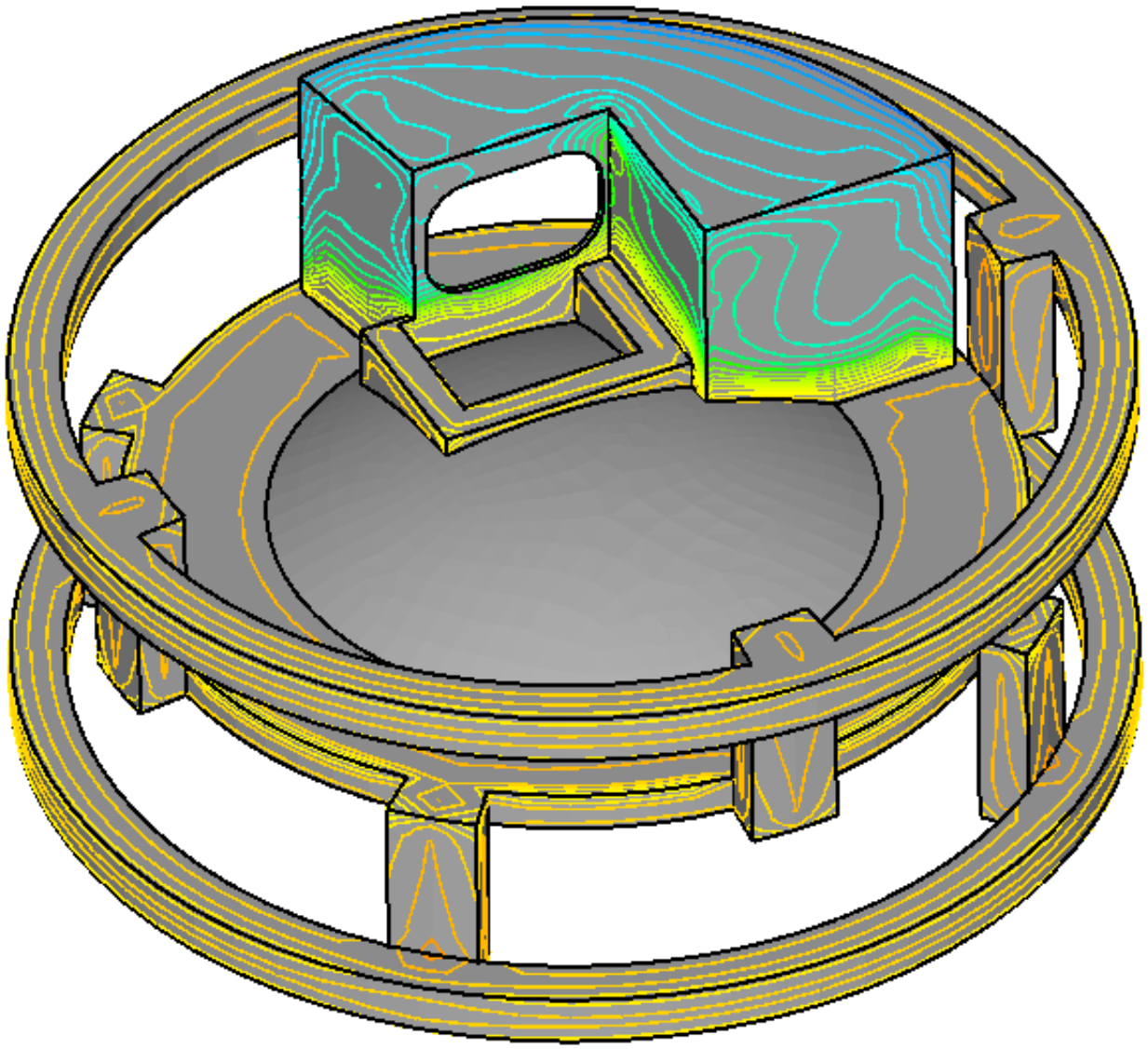


Figure 4.16

4.2.5 Iso Surface

This visualization method is for node based data. It draws surfaces of constant value. Up to 5 surfaces may be specified with each module. The *Iso Component* scroll window is used to select which data component is used to calculate the isosurface and the *Map Component* scroll window is used to select which data component is color mapped onto the surface. The *iso level* sliders set the values for each isosurface and the toggle to the right of each slider turns that surface on or off.

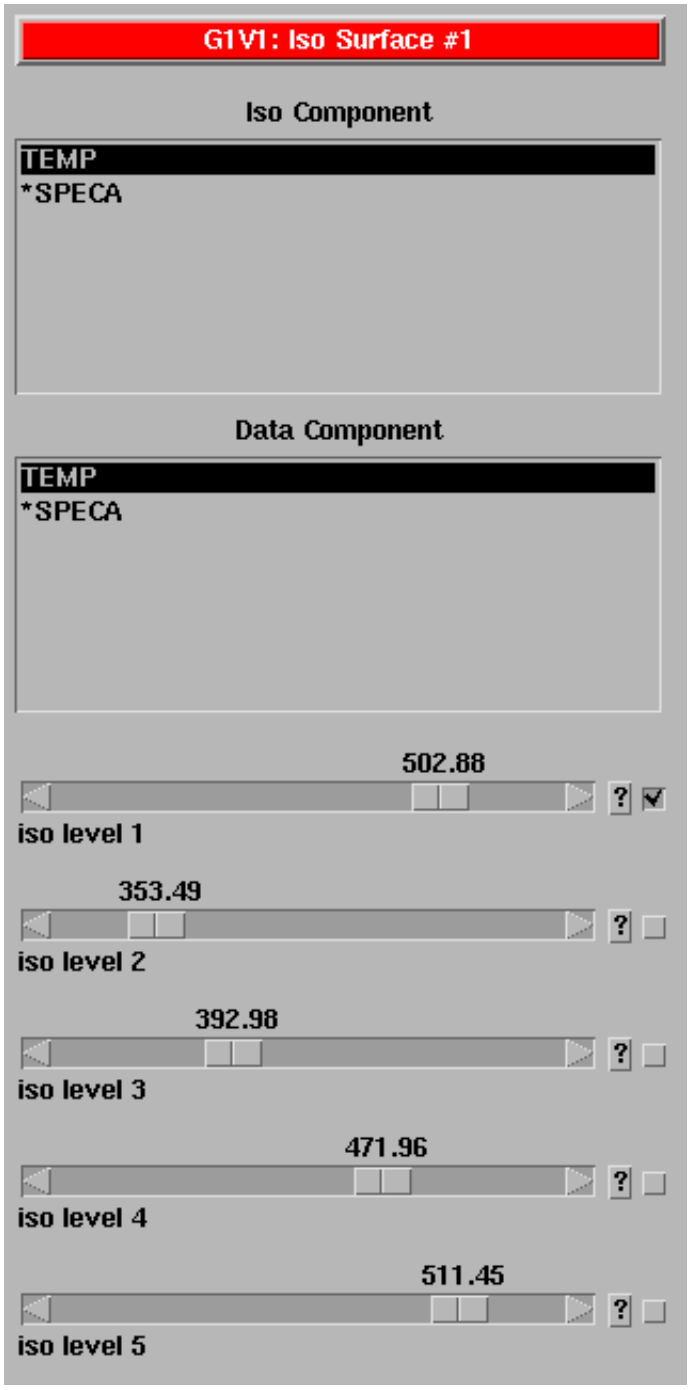


Figure 4.17 Control panel for “Iso Surface”

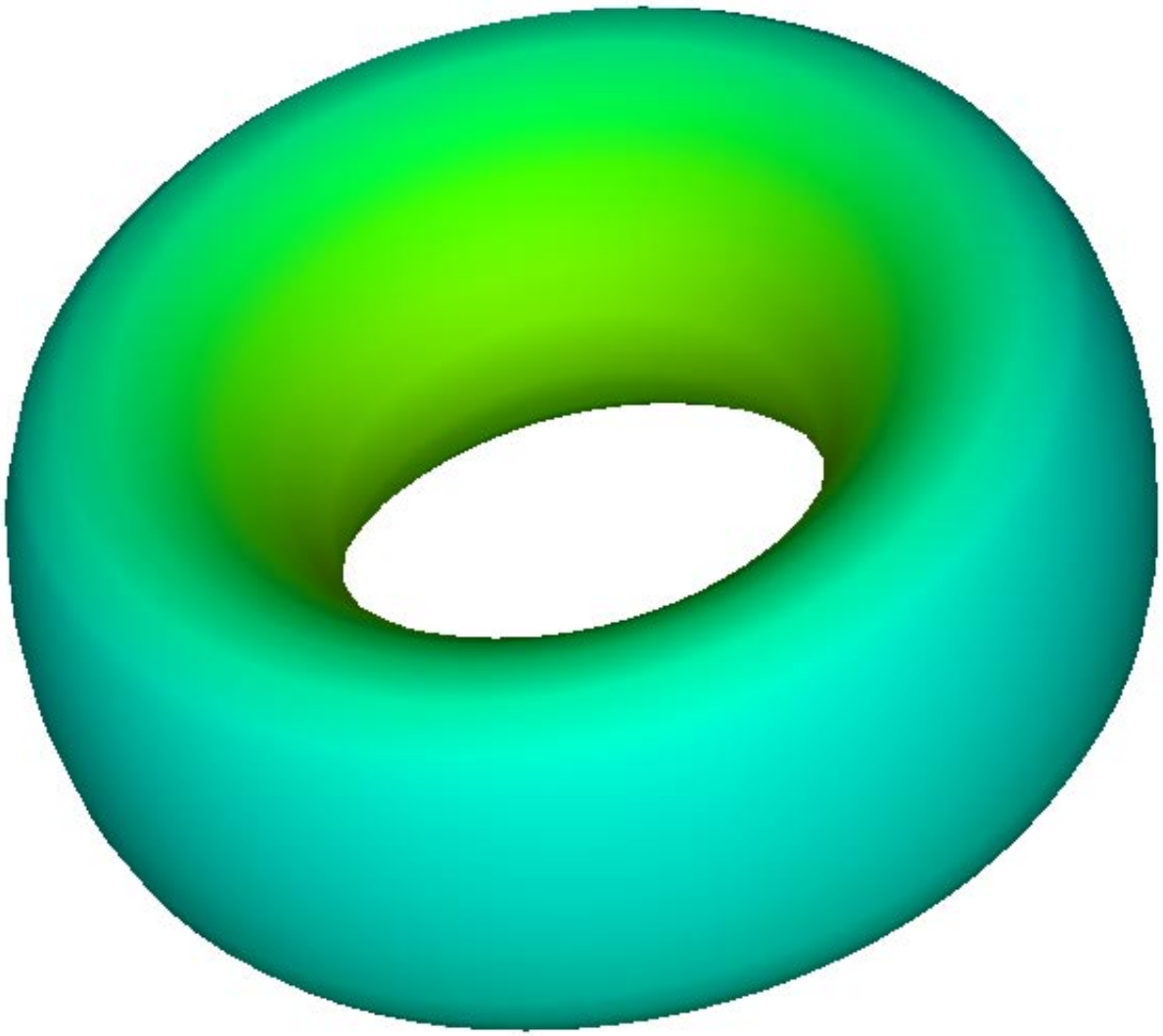


Figure 4.18

4.2.6 Iso Volume

This visualization method is for node based data. It draws volumes bounded by two isosurfaces. The *Iso Component* scroll window is used to select which data component is used to calculate the isosurfaces and the *Map Component* scroll window is used to select which data component is color mapped onto the surface of the volume. The *min level* and *max level* sliders set the values for each isosurface.

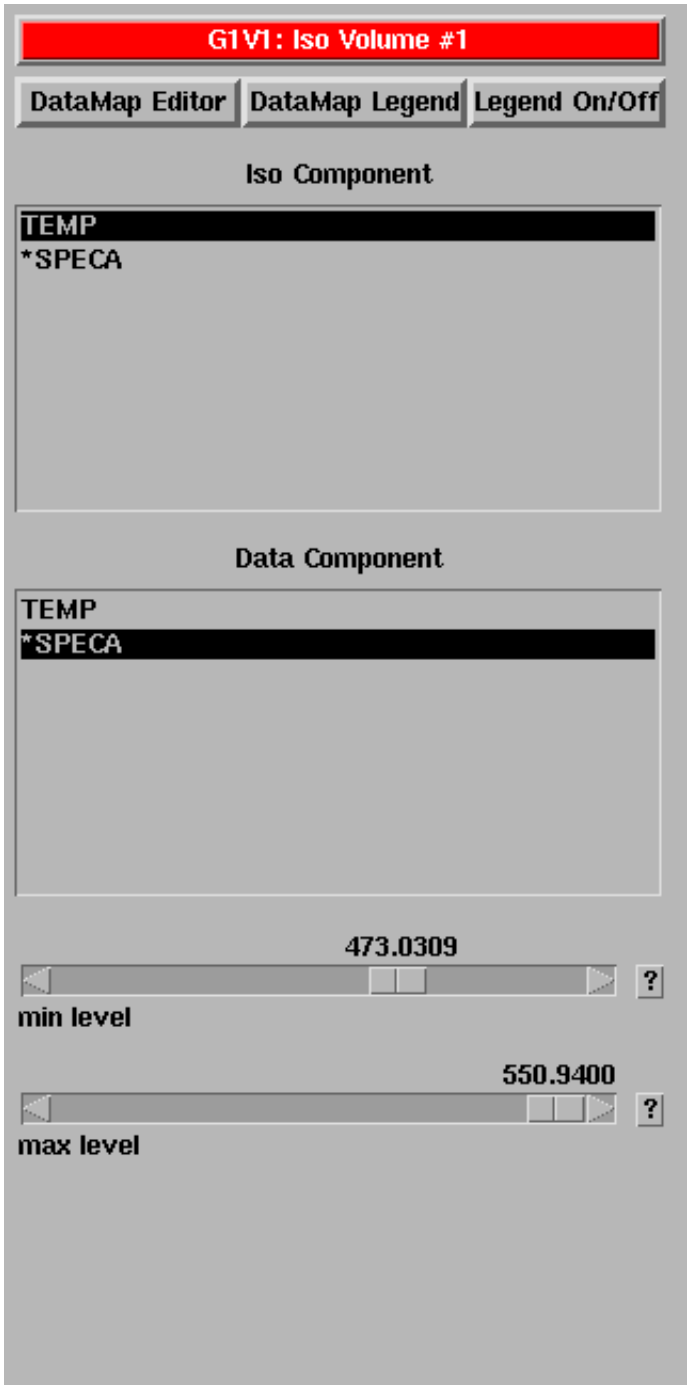


Figure 4.19 Control panel for “Iso Volume”

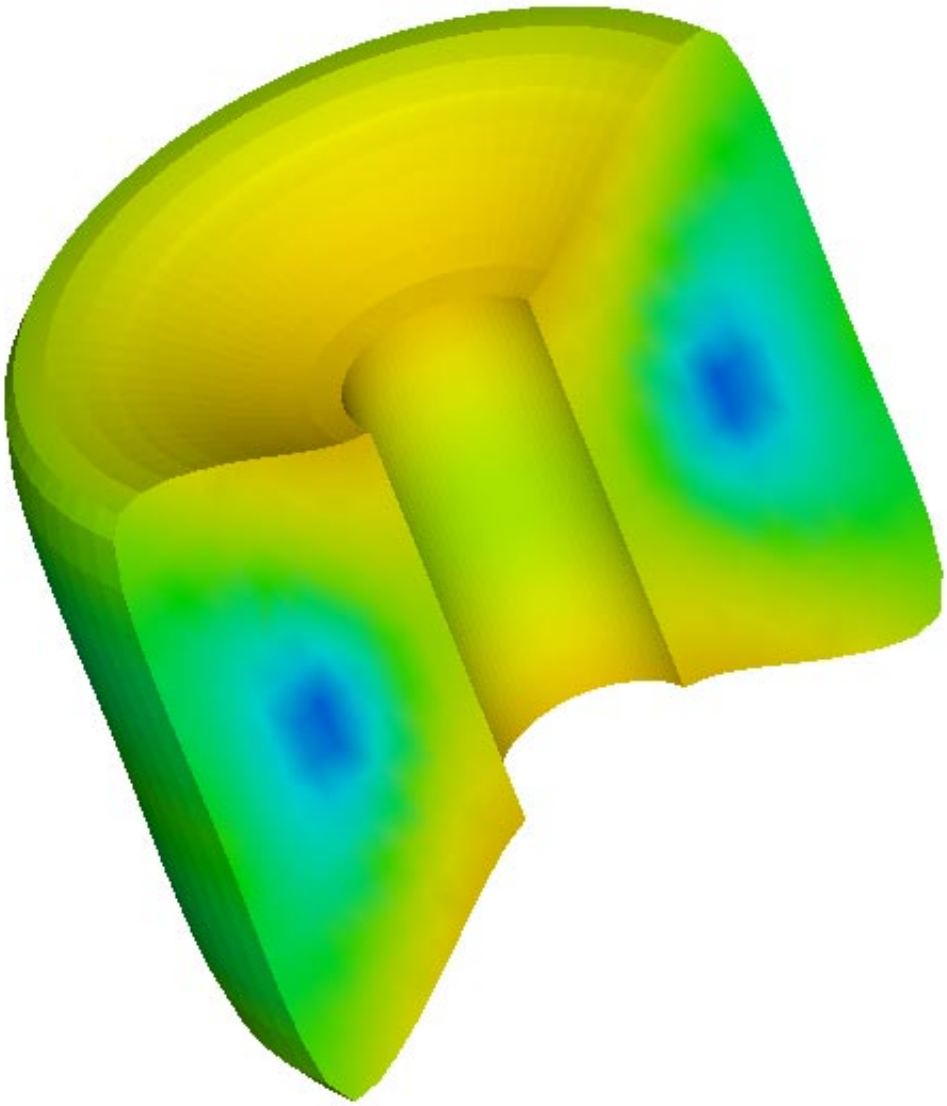


Figure 4.20 Example of “Iso Volume”

4.2.7 Paint Cells

This visualization method is for element based data. It paints (colors) the entire element according to that element's color-mapped value. Mesh lines and/or edge lines can be displayed and the line's attributes can also be adjusted. The *Data Map Editor* button will activate a popup window to control the data-to-color mapping and the *Data Map Legend* button will activate a popup window to display a legend in the view window. The *Legend On/Off* button will toggle the legend display on or off.

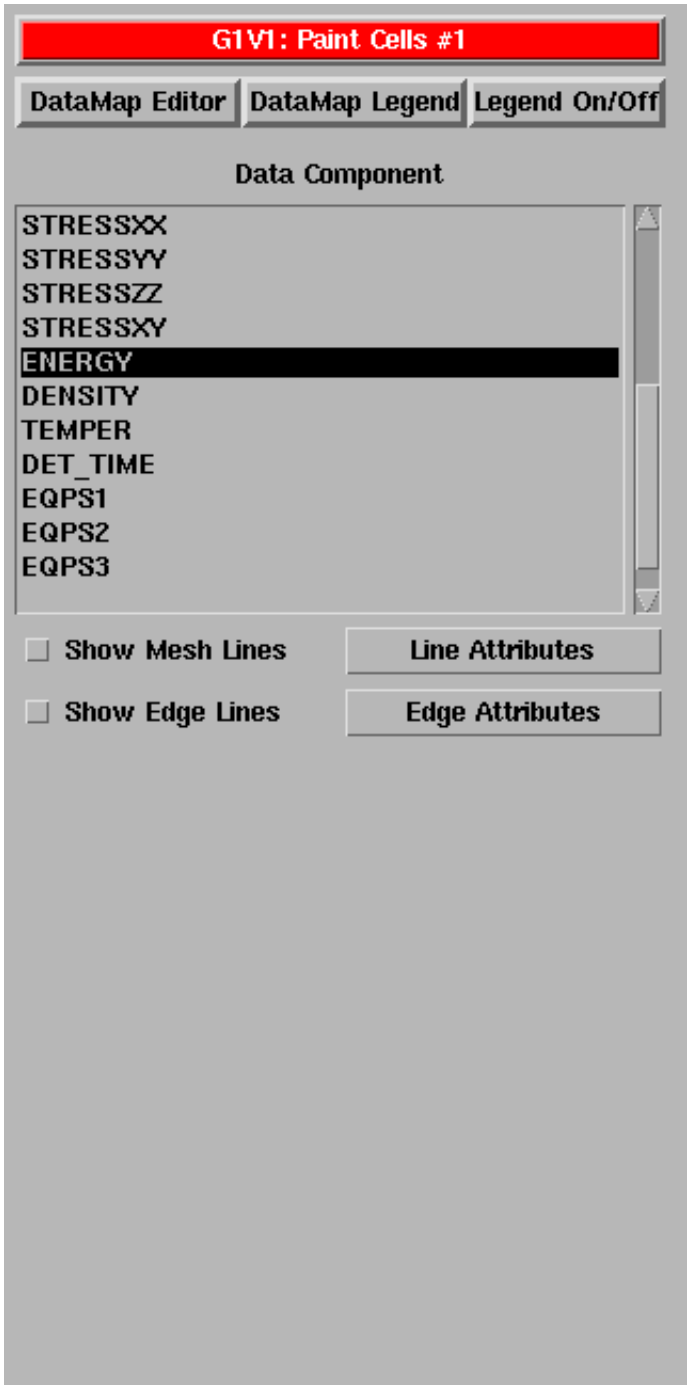


Figure 4.21 Control panel for "Paint Cells"

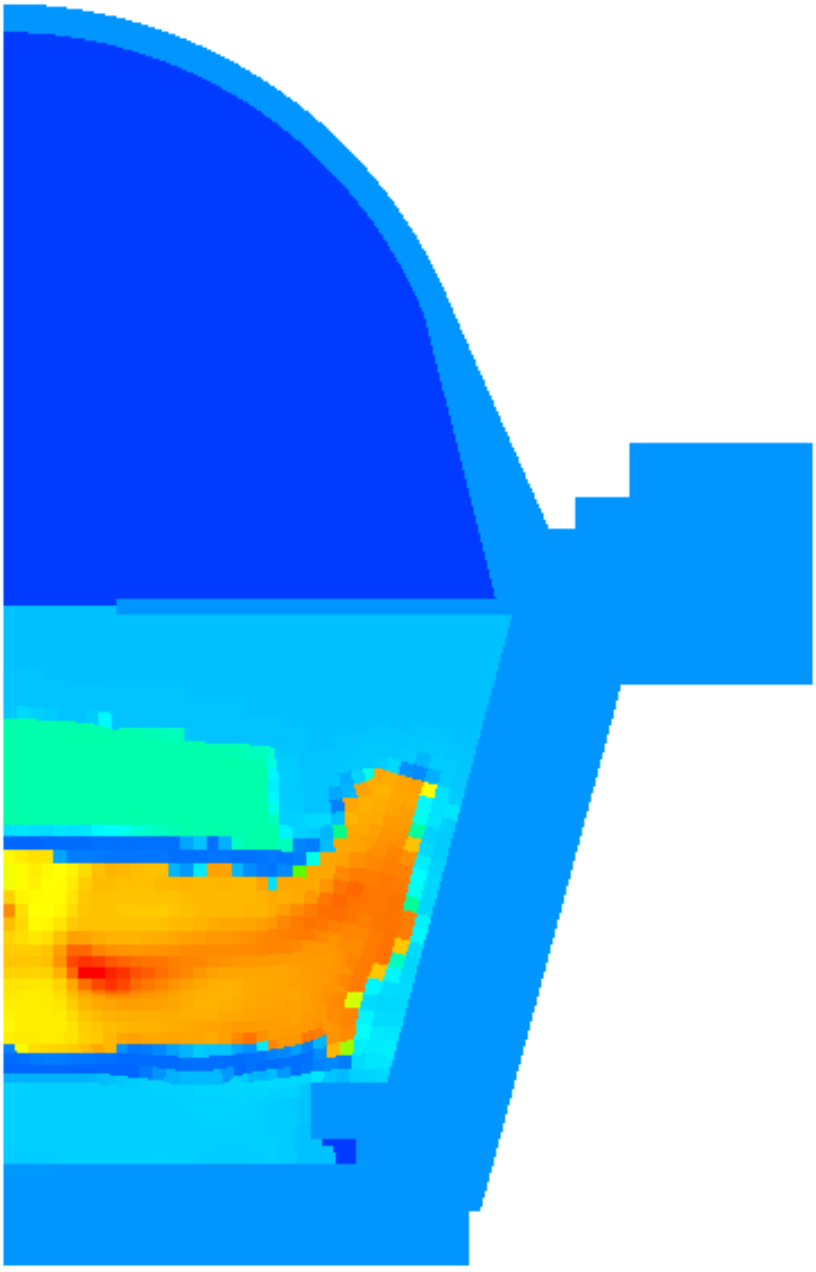


Figure 4.22 Sample “Paint Cells” viz method

4.2.8 Multi-Material Map

This visualization method is for element based data. It constructs a multimaterial map from the selected data components and then colors (paints) the entire element according to that element's colormapped value. The individual materials are colored from blue to red based upon their selection order in the selection box. Mesh linesand/or edge lines can be displayed and the line's attributes can also be adjusted. The *Data Map Editor* button will activate a popup window to control the data-to-color mapping and the *Data Map Legend* button will activate a popup window to display a legend in the view window. The *Legend On/Off* button will toggle the legend display on or off.

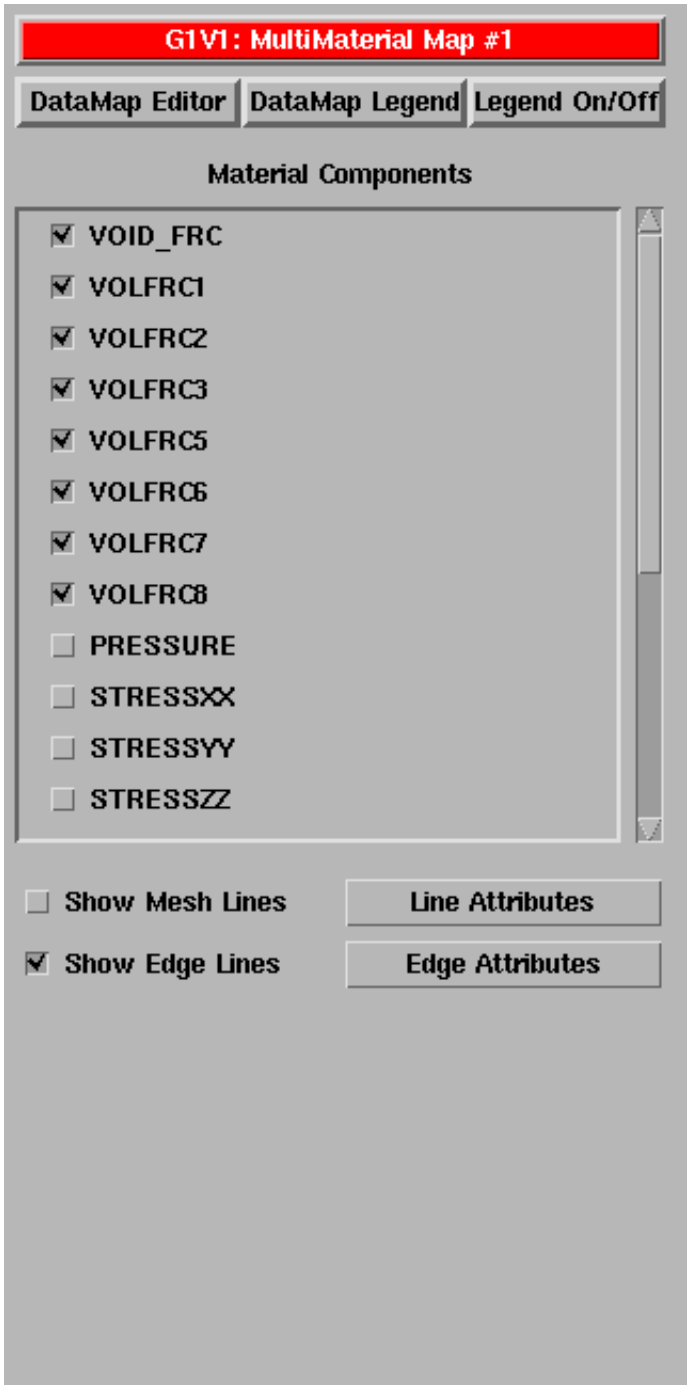


Figure 4.23 Control panel for "Multimaterial Map"

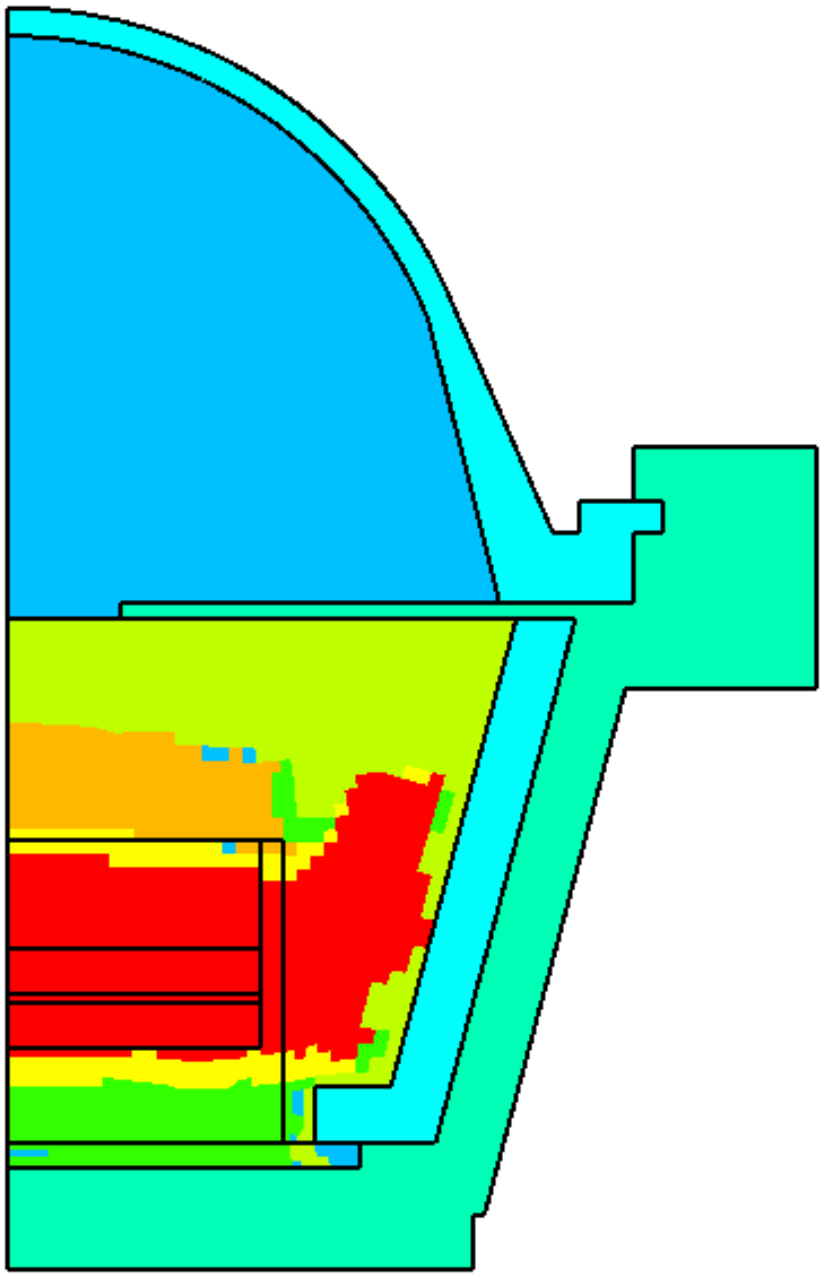


Figure 4.24 Sample of “Multi-Material” viz method

4.2.9 Particle Field

This visualization method is for particle fields stored in a specific exodusII format. It constructs a particle field, automatically applying particle displacements and radii values. The field is represented with a sphere at each point. Setting the glyph scale value to 0.0 causes a point to be rendered at each particle position instead of a sphere. The *Data Map Editor* button will activate a popup window to control the data-to-color mapping and the *Data Map Legend* button will activate a popup window to display a legend in the view window. The *Legend On/Off* button will toggle the legend display on or off.

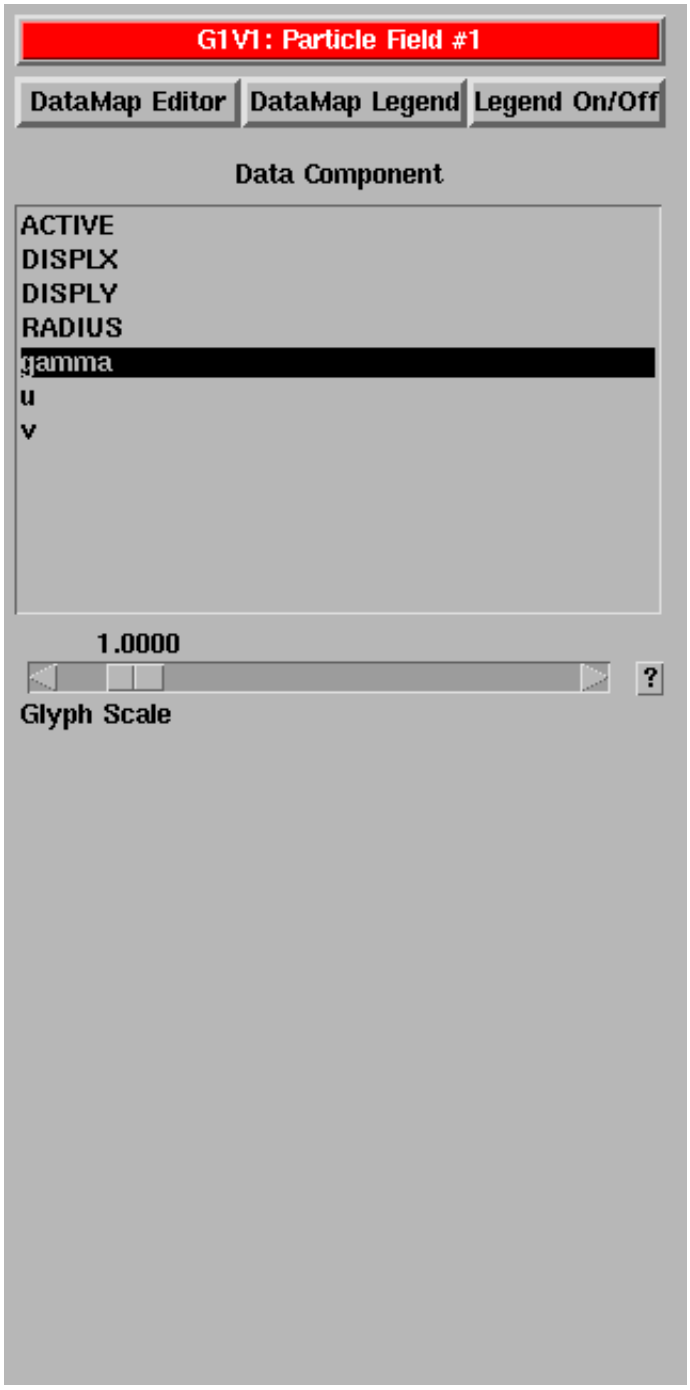


Figure 4.25 Control panel for “Particle Field”

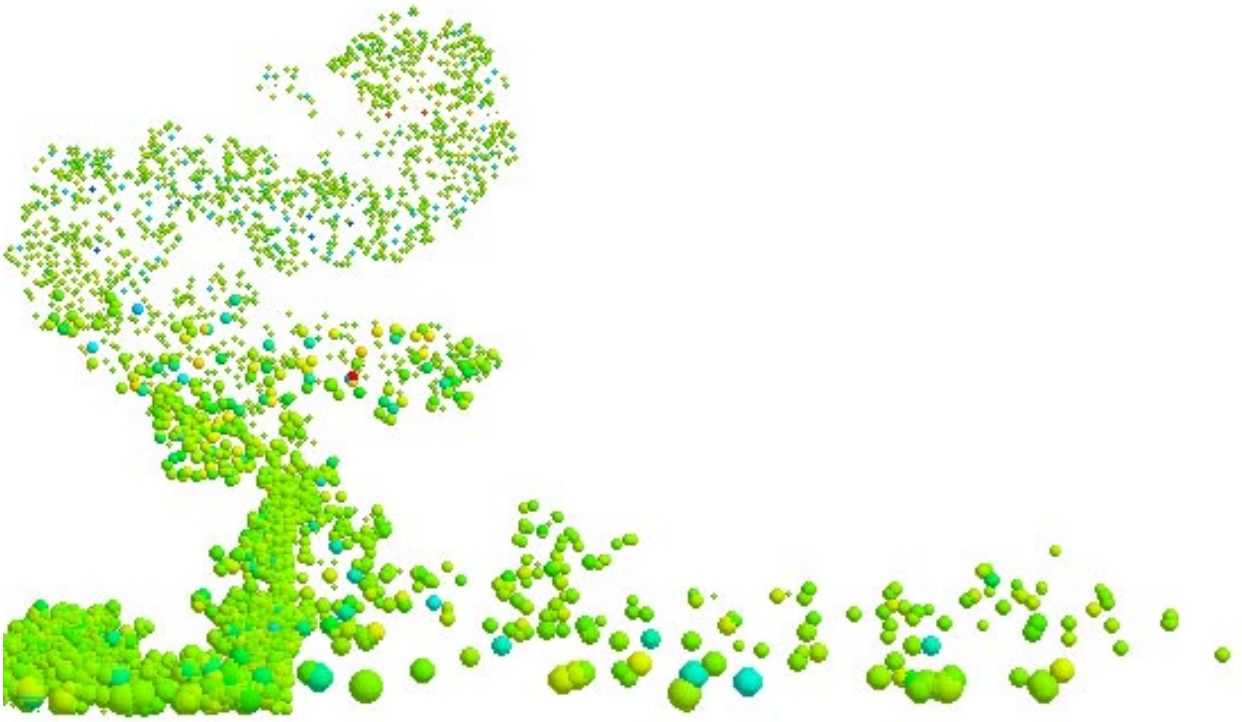


Figure 4.26 Sample of “Particle Field” viz method

4.3 Vector Visualization Methods

4.3.1 Hedge Hog

This method creates vector plots of vector data. The *Construct Vector* button widget will activate a popup window to allow the user to select the components that make up the vector. The *Glyph* popup menu selects what style of glyph will be drawn and the *Probe* popup menu selects where the glyphs will be drawn (e.g. at the nodes, on a plane, on a line, etc.). The glyphs can be colored according to the magnitude of the vector by selecting the *Color Glyph* toggle widget. Otherwise, the glyph color is set according to the HSV color model dial widgets. The size the glyphs is scaled according to the magnitude of the vector or the may all be set to the same size by selecting the *Normalize Glyph* toggle widget. The glyph size may be further sized by the *Glyph Scale* widget. This is a linear scale unless the *Power of 10 scaling* toggle widget is selected. In this case the scale value is the value of the exponent in:

$$\text{(actual scale value)} = 10^{(\text{scale value})}$$

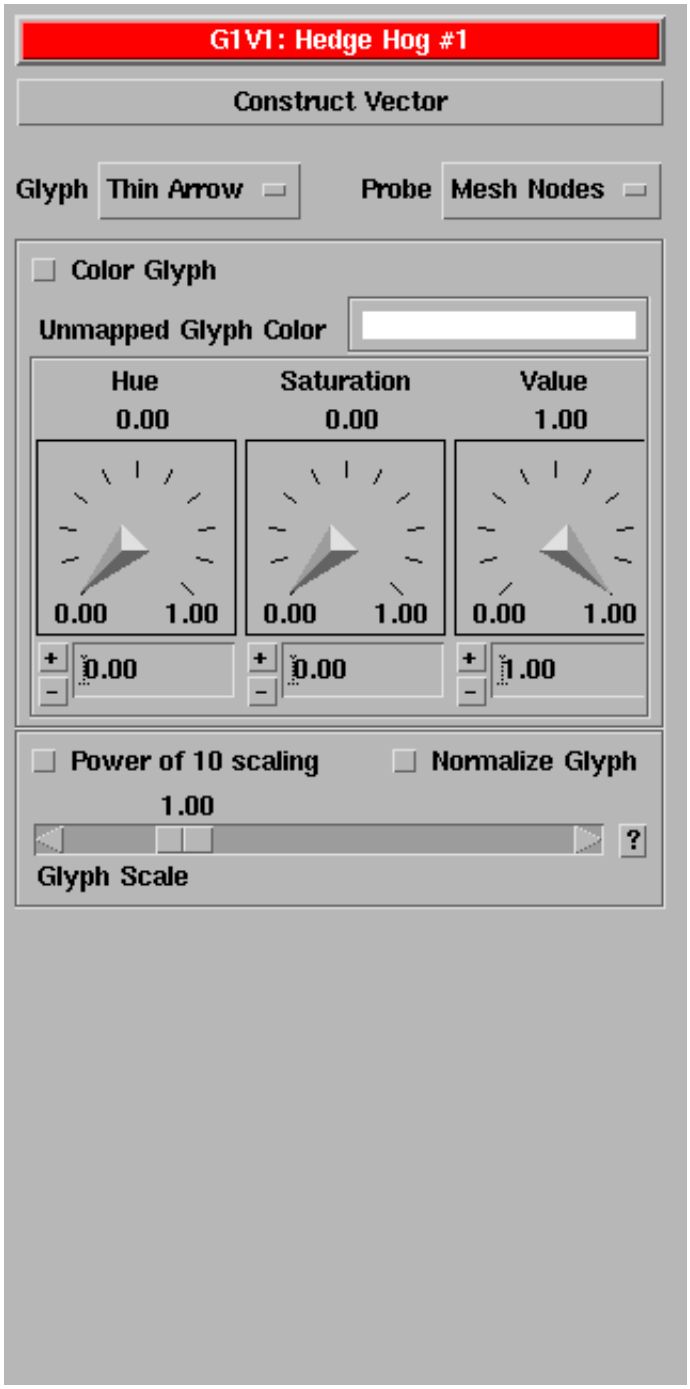


Figure 4.27 Control Panel for “Hedge Hog”

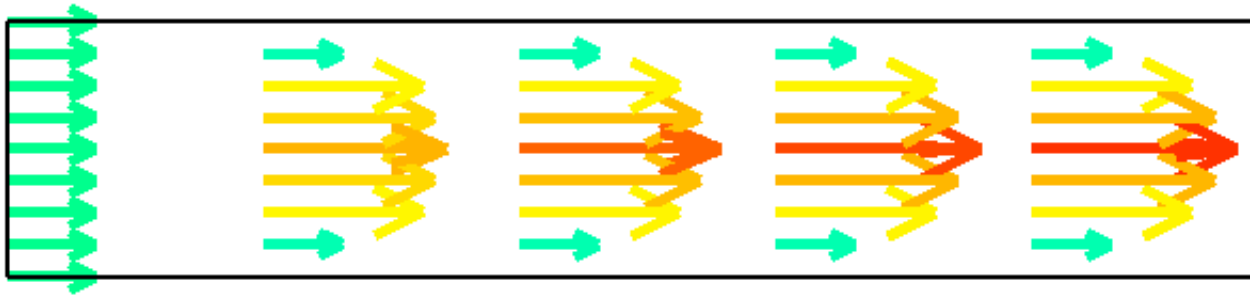


Figure 4.28 Example of “Hedge Hog”

4.3.2 Streamlines

This method creates streamlines or streamribbons by integrating a velocity vector field. The *Construct Vector* button widget will activate a popup window to allow the user to select the components that make up the vector. The *Direction* popup menu is used to specify the integration direction - forward or backward. The *N-segments* slider widget is used to specify how many integration steps are taken within each element to compute the streamline path. The *Max Segments* slider widget sets the total number of integration steps. The *Order* slider widget specifies the integration order. Higher orders are more accurate, but execute more slowly. When a streamline falls below the value specified by the *Min Velocity* slider widget, integration for that streamline stops. The *Color* toggle widget specifies whether or not the streamlines (ribbons) are colormapped. The *Ribbons* toggle widget specifies streamribbons instead of streamlines. The *Ribbon Width* slider widget specifies the streamribbon width and the *Ribbon Angle* slider widget specifies the initial streamribbon angle. The *Probe Type* popup menu species where the initial integration points will be - either at the mesh nodes, on an arbitrary plane passing through the model, or on an arbitrary line passing through the model.

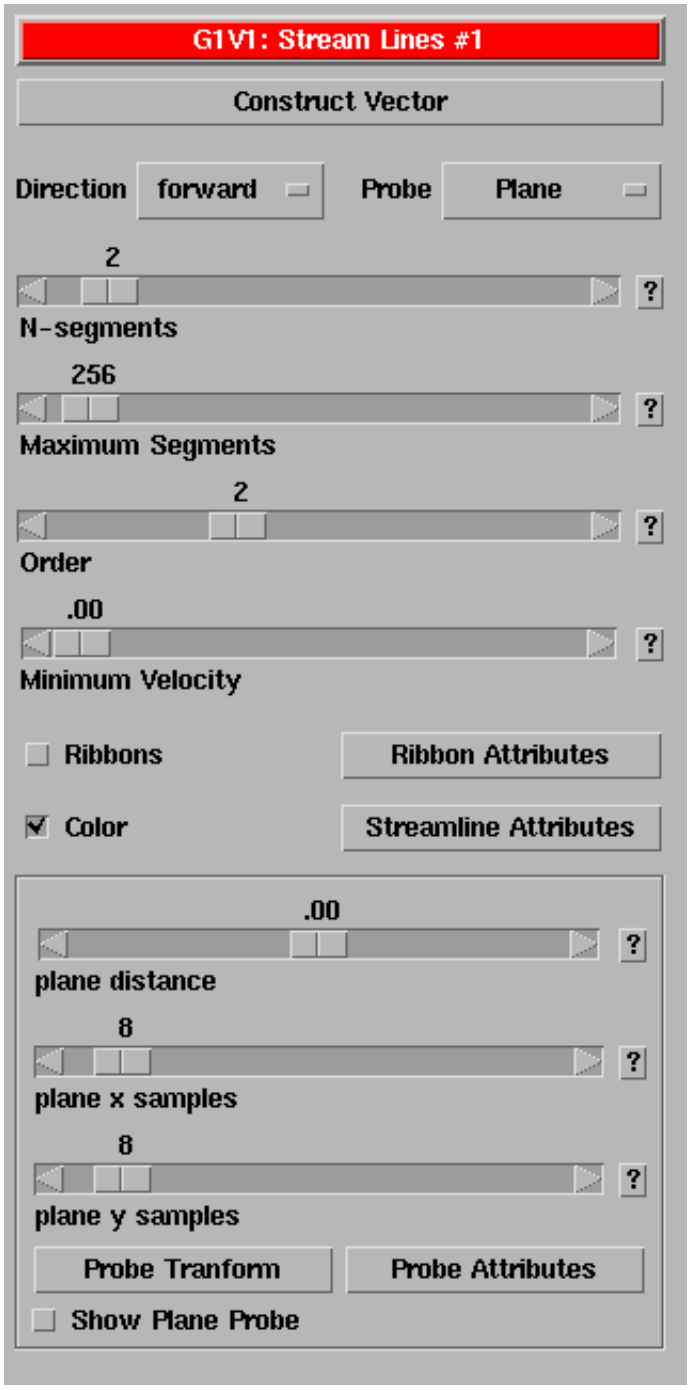


Figure 4.29 Control Panel for “Streamlines”

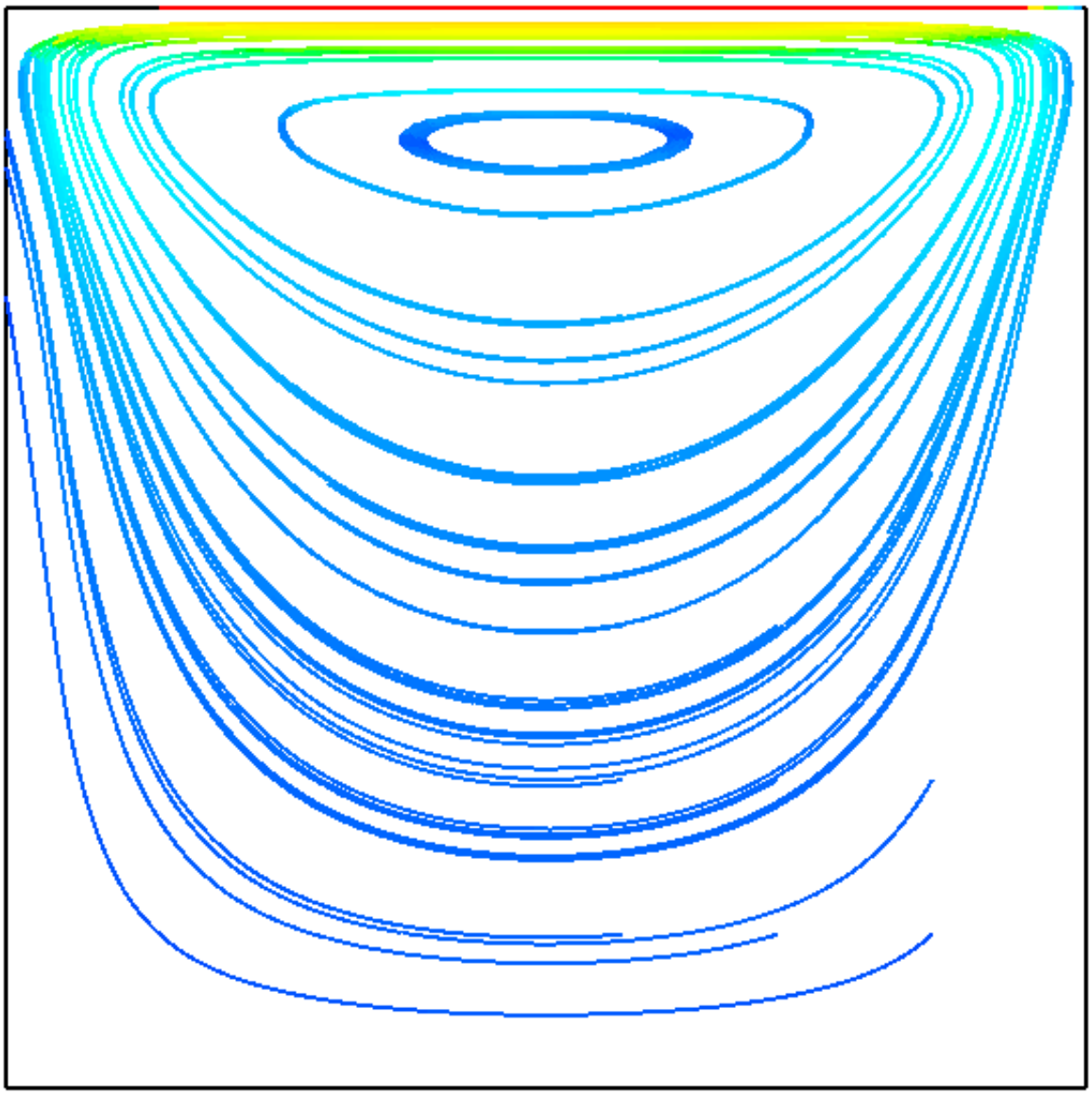


Figure 4.30 Example of “Streamlines”

4.3.3 Advect Particles

This method releases massless particles into a velocity field and advects them along streamlines. The *Construct Vector* button widget will activate a popup window to allow the user to select the components that make up the vector. The *N-segments* slider widget is used to specify how many integration steps are taken within each element to compute the streamline path. The *Max Segments* slider widget sets the total number of integration steps. The *Order* slider widget specifies the integration order. Higher orders are more accurate, but execute more slowly. When a streamline falls below the value specified by the *min velocity* slider widget, integration for that streamline stops. The *Direction* popup menu is used to specify the integration direction - forward or backward. The *Glyph Style* popup menu is used to specify the geometry of the glyphs used to represent the particles. The *Glyph Normalize* toggle widget specifies whether or not the glyphs are scaled according to the velocity magnitude. The *Start Time*, *End Time*, and *Step* slider widgets setup the time integration parameters. The *Run* toggle widget begin the advection, the *Reset Time* toggle resets the time value, and the The *Cycle* toggle widget puts the particle advection in a cyclic loop. The *Probe* popup menu species where the initial integration points will be - either at the mesh nodes, on an arbitrary plane passing through the model, or on an arbitrary line passing through the model.

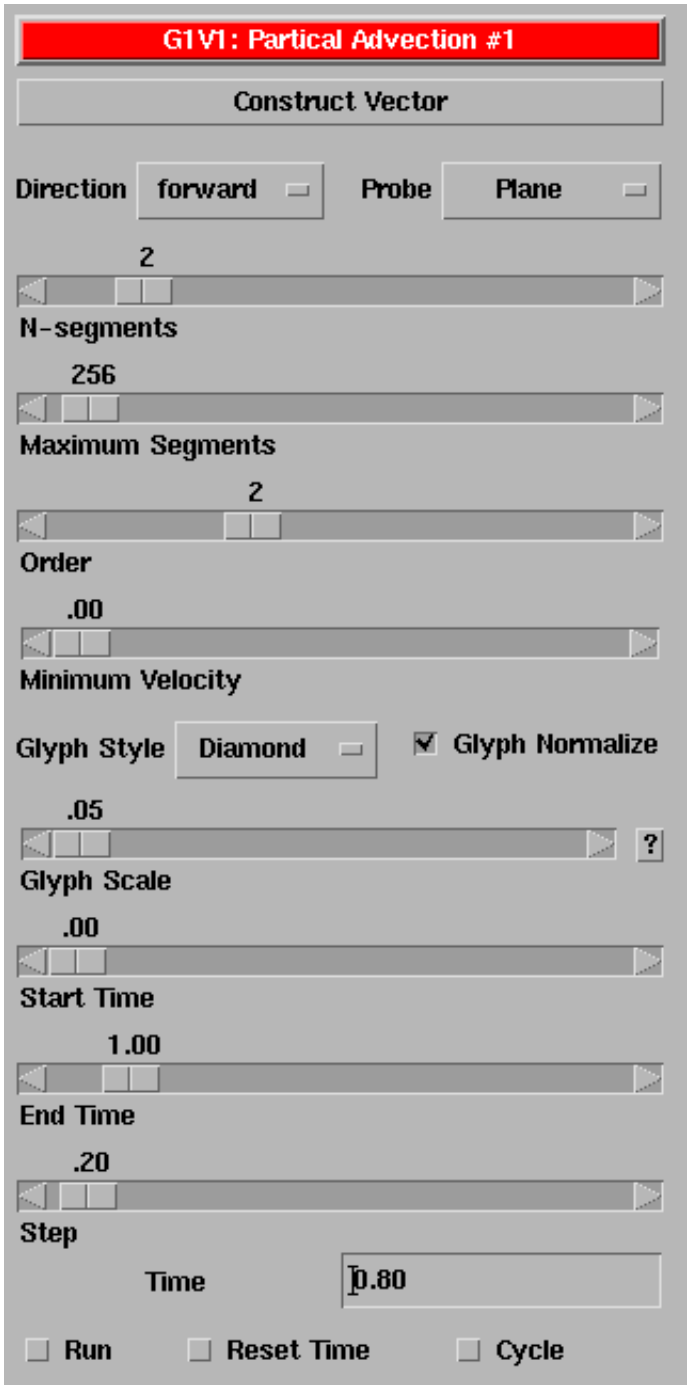


Figure 4.31 Control panel for “Advect Particles”

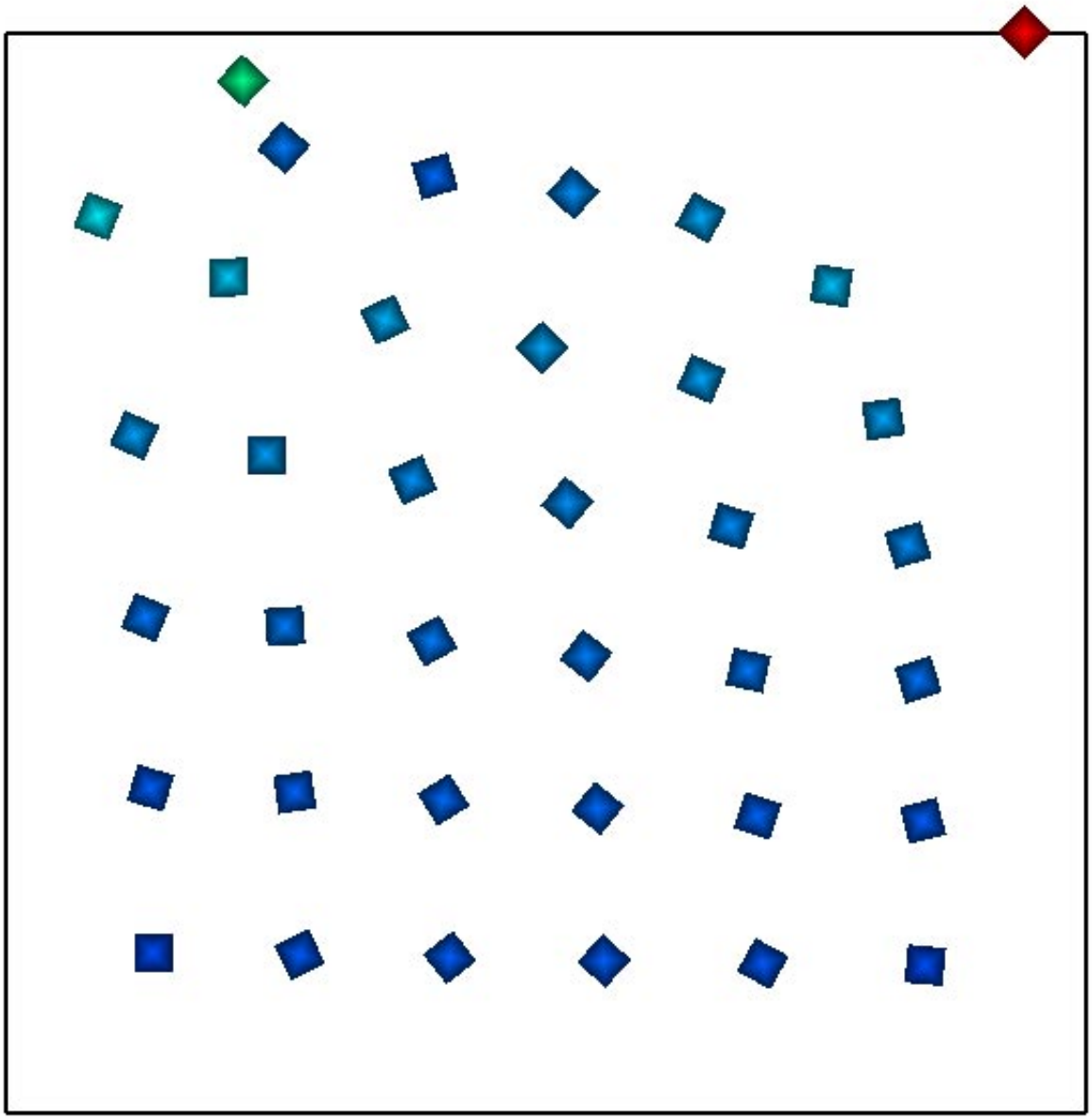


Figure 4.32

4.4 X-Y Plotting Methods

4.4.1 Data vs. Distance Plot

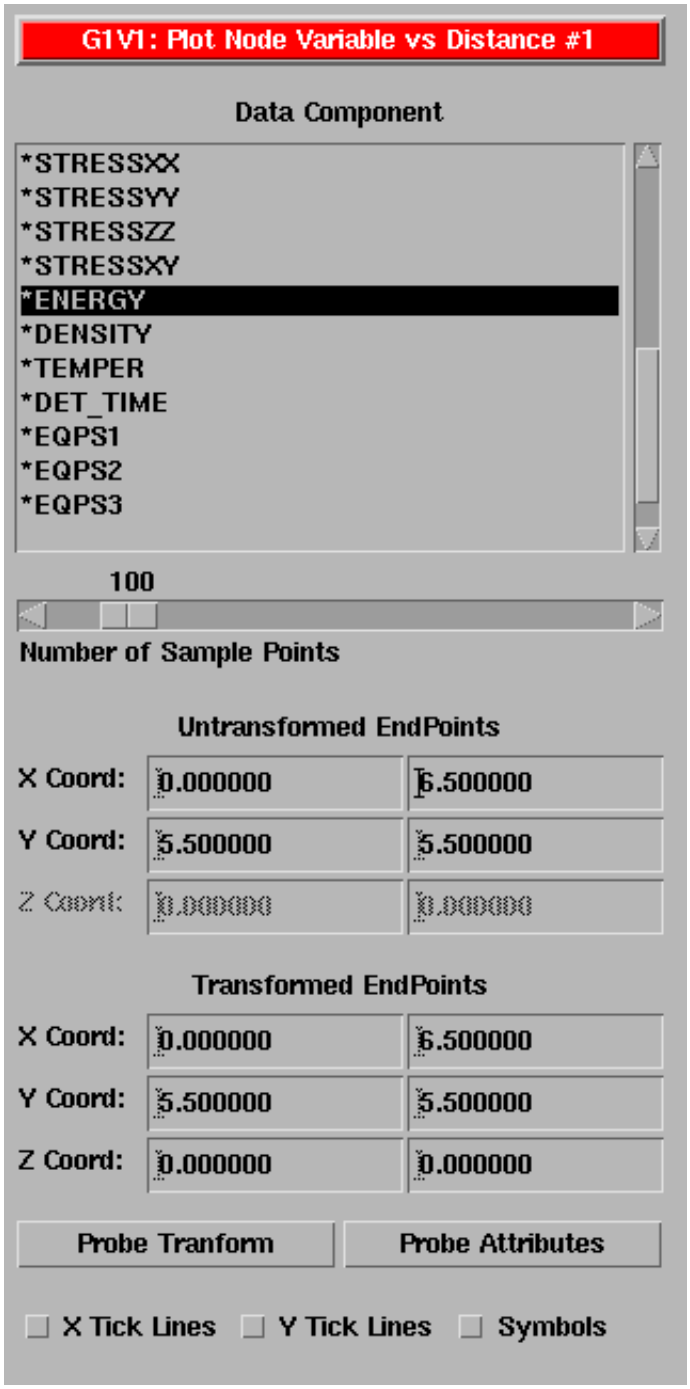


Figure 4.33 Control panel for “Node Plot”

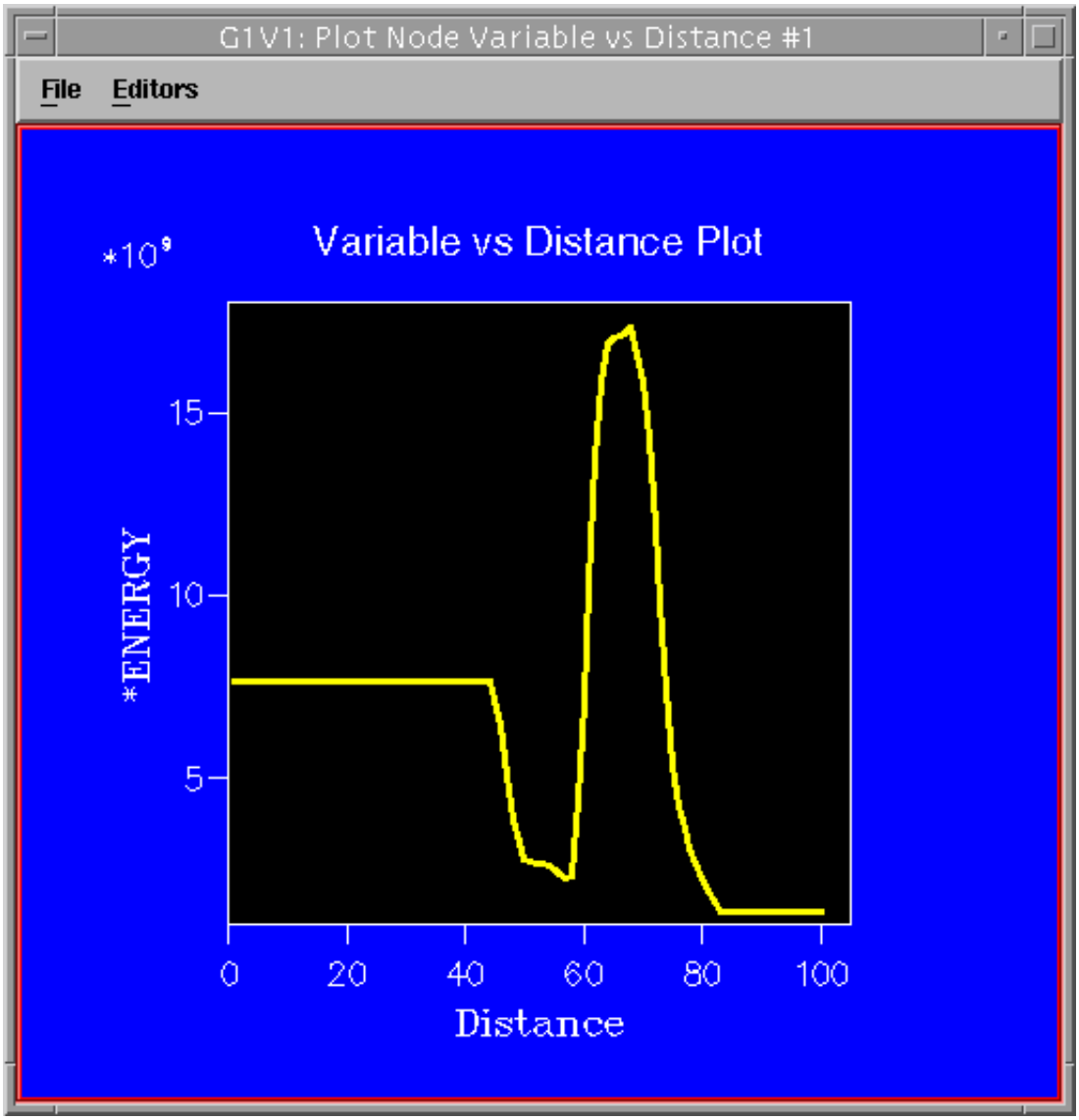


Figure 4.34 Example of “Node Variabl vs Distance Plot”

4.5 Query Methods

4.5.1 Data Query

This visualization module interactively shows numeric data values nodal data values at a probe location. The *Data Component* selection box widget is used to select what data value is to be queried. The *Probe Normalize* toggle widget selects whether or not the probe size is scaled according to the data at its location. The *probe scale* slider widget scales the probe size. The *Probe Transform Popup* button displays a transformation control window to set the probe position. Or the mouse can be used to select the probe position. Clicking the left-mouse button in the viewer window while pressing the `control` key selects a new probe position. The coordinates of the probe position and the data value are displayed.

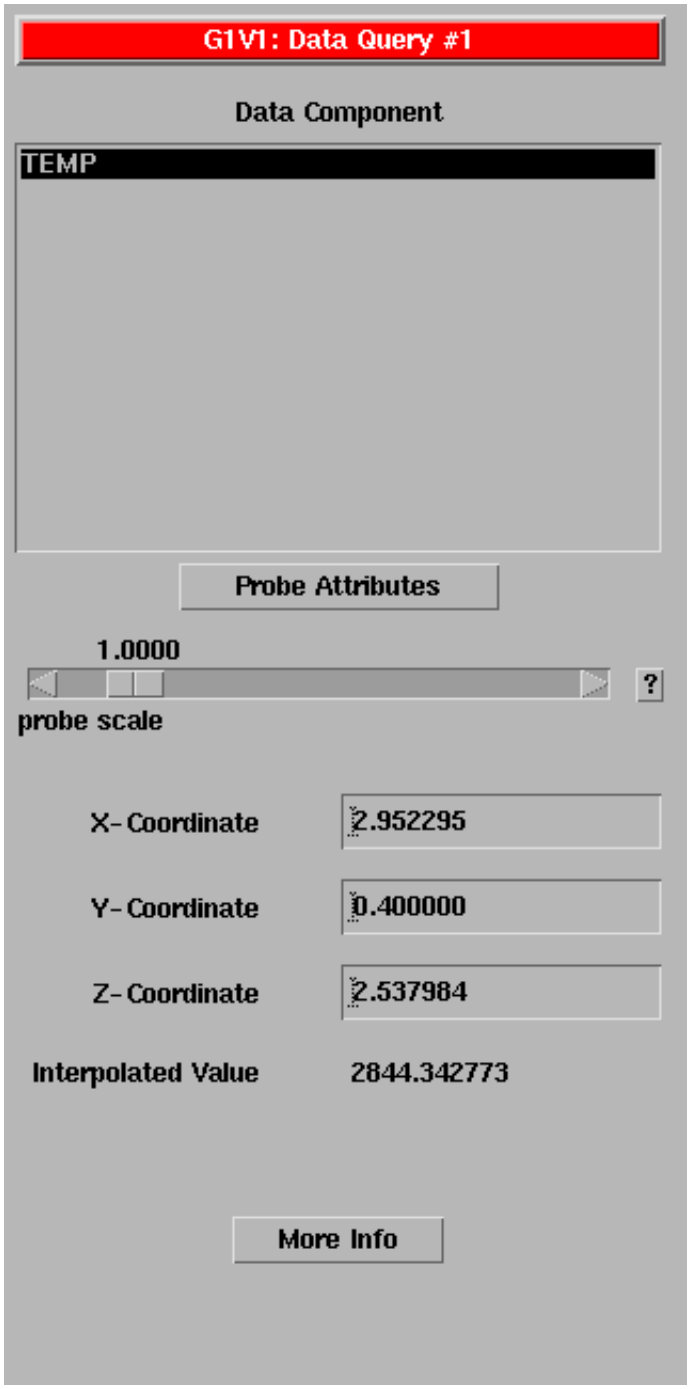


Figure 4.35 Control panel for “Data Query”

4.6 Annotation Methods

4.6.1 Exodus Time Stamp

This visualization method displays the time plane value from an ExodusII database. The position of the annotation is controlled by the *X*, *Y*, and *Z Position* typein widgets. Valid values for these coordinates are +/- 1.0 The color of the text is controlled by the HSV color model dial widgets. The horizontal and vertical alignment of the text in relation to the position coordinate is controlled by the *Align Horizontal* and *Align Vertical* popup menus. The *Bounds* toggle widget will draw a bounding box around the text and the *Underline* toggle widget will draw a line under the text. The *Format* typein widget is used to control the format of the displayed text string. C style formatting convention is used here. The *Font* typein widget specifies the font to be used. The font is specified using the X-windows convention. The `xfontsel` utility can be used to help select the font string. The *Time Scale Factor* typein widget is used to supply a value by which the time value is multiplied. This product is then supplied to the format string. This scale factor is useful for units conversion of the time value such as converting second to microseconds.

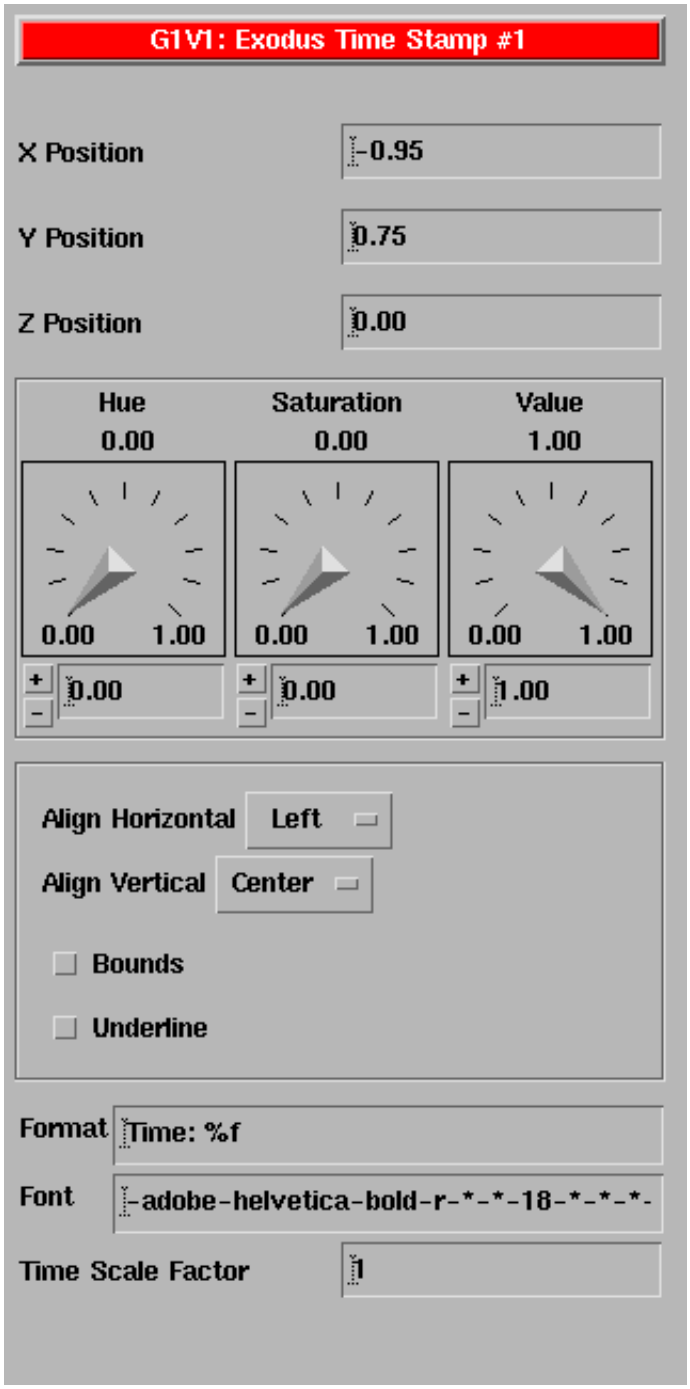


Figure 4.36 Control panel for “Exodus Time Stamp”

4.6.2 Global Variable Annotation

This visualization method displays the value of a global variable in an annotation in the active viewer. The value of the global variable is avaible for formatting accoding to the format string. The variable is selected in the *Global Variable* scroll widget. The position of the annotation is controlled by the *X*, *Y*, and *Z Position* typein widgets. Valid values for these coordinates are +/- 1. The color of the text is controlled by the HSV color model dial widgets. The horizontal and vertical alignment of the text in relation to the position coordinate is controlled by the *Align Horizontal* and *Align Vertical* popup menus. The *Bounds* toggle widget will draw a bounding box around the text and the *Underline* toggle widget will draw a line under the text. The *Format* typein widget is used to control the format of the displayed text string. C style formatting convention is used here. The *Font* typein widget specifies the font to be used. The font is specified using the X-windows convention. The `xfontsel` utility can be used to help select the font string. The *Scale Factor* typein widget is used to supply a value by which the time value is multiplied. This product is then supplied to the format string. This scale factor is useful for units conversion.

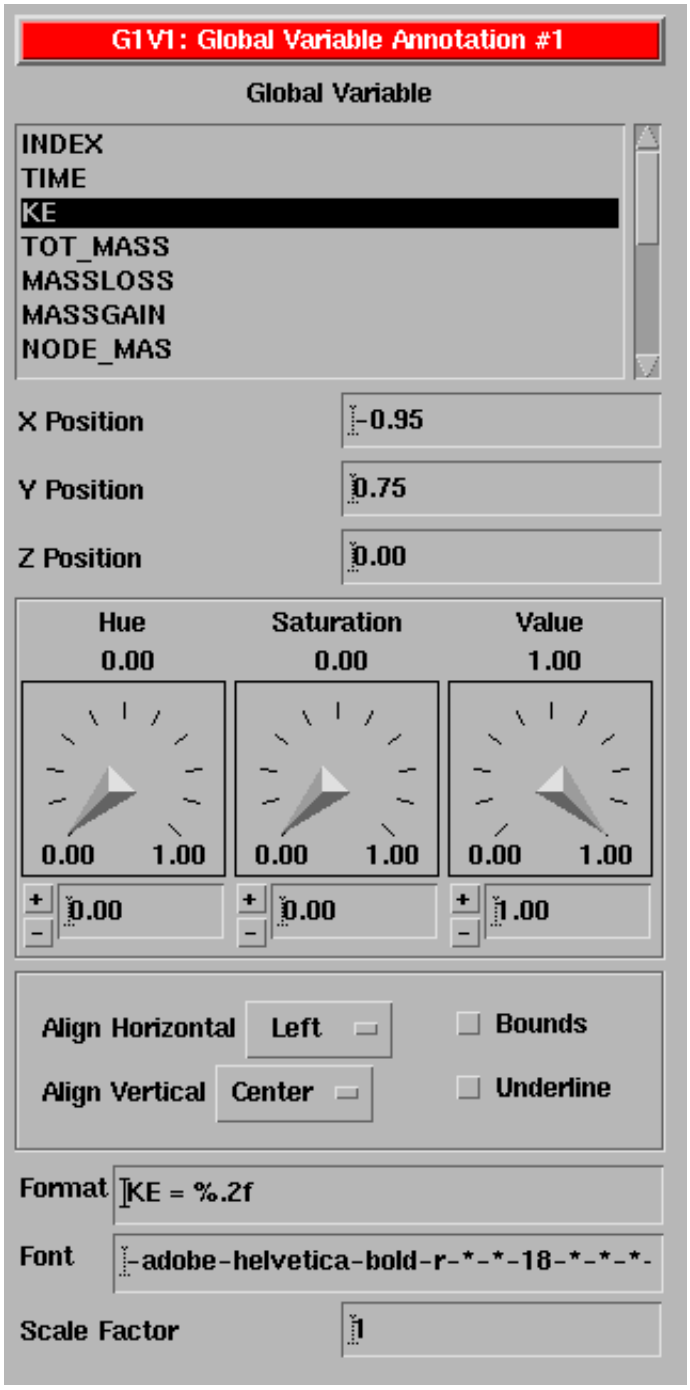


Figure 4.37 Control panel for “Global Vraiable Annotation”

4.6.3 Nodal Variable Annotation

This visualization method displays the value of a nodal variable at a particular node in an annotation in the active viewer. The value of the nodal variable is avaible for formatting accoding to the format string. The variable is selected in the *Nodal Variable* scroll widget. The node number is set in a typein widget. The position of the annotation is controlled by the *X*, *Y*, and *Z Position* typein widgets. Valid values for these coordinates are +/- 1. The color of the text is controlled by the HSV color model dial widgets. The horizontal and vertical alignment of the text in relation to the position coordinate is controlled by the *Align Horizontal* and *Align Vertical* popup menus. The *Bounds* toggle widget will draw a bounding box around the text and the *Underline* toggle widget will draw a line under the text. The *Format* typein widget is used to control the format of the displayed text string. C style formatting convention is used here. The *Font* typein widget specifies the font to be used. The font is specified using the X-windows convention. The `xfontsel` utility can be used to help select the font string. The *Scale Factor* typein widget is used to supply a value by which the variable's value is multiplied. This product is then supplied to the format string. This scale factor is useful for units conversion.

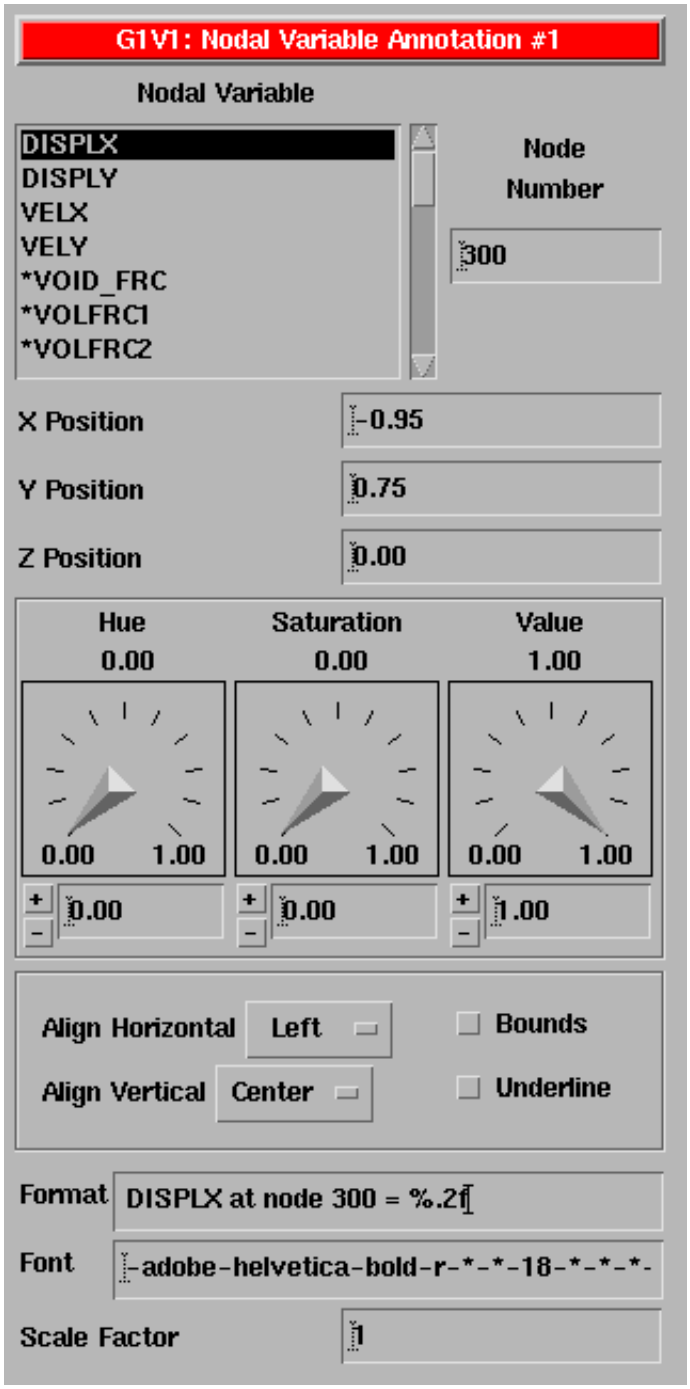


Figure 4.38 Control panel for "Nodal Variable Annotation"

4.6.4 Element Variable Annotation

This visualization method displays the value of a element variable at a particular element in an annotation in the active viewer. The value of the element variable is avaible for formatting accoding to the format string. The variable is selected in the *Element Variable* scroll widget. The element number is set in a typein widget. The position of the annotation is controlled by the *X*, *Y*, and *Z Position* typein widgets. Valid values for these coordinates are +/- 1. The color of the text is controlled by the HSV color model dial widgets. The horizontal and vertical alignment of the text in relation to the position coordinate is controlled by the *Align Horizontal* and *Align Vertical* popup menus. The *Bounds* toggle widget will draw a bounding box around the text and the *Underline* toggle widget will draw a line under the text. The *Format* typein widget is used to control the format of the displayed text string. C style formatting convention is used here. The *Font* typein widget specifies the font to be used. The font is specified using the X-windows convention. The `xfontsel` utility can be used to help select the font string. The *Scale Factor* typein widget is used to supply a value by which the variable's value is multiplied. This product is then supplied to the format string. This scale factor is useful for units conversion.

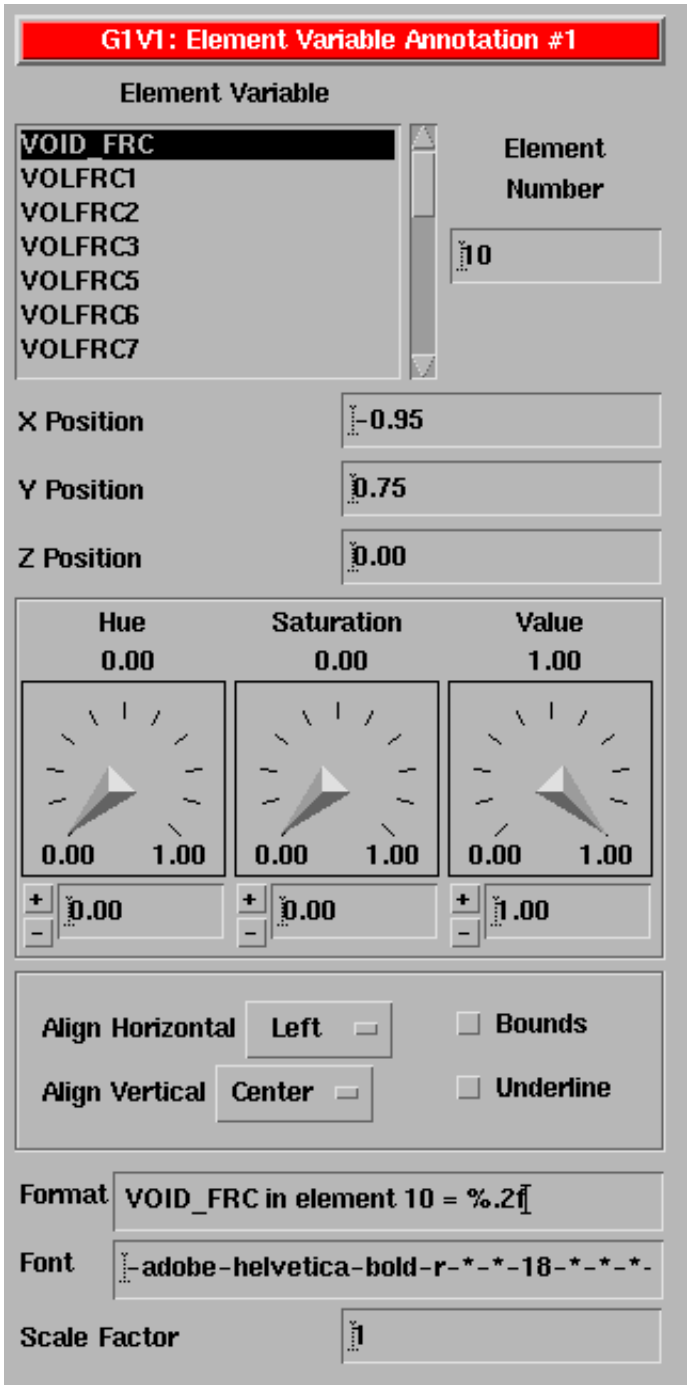


Figure 4.39 Control panel for "Element Variable Annotation"

4.6.5 User Annotations

This visualization method displays user annotation in the active viewer. The position of the annotation is controlled by the *X*, *Y*, and *Z Position* typein widgets. Valid values for these coordinates are +/- 1. The color of the text is controlled by the HSV color model dial widgets. The horizontal and vertical alignment of the text in relation to the position coordinate is controlled by the *Align Horizontal* and *Align Vertical* popup menus. The *Bounds* toggle widget will draw a bounding box around the text and the *Underline* toggle widget will draw a line under the text. The *Format* typein widget is used to control the format of the displayed text string. C style formatting convention is used here. The *Font* typein widget specifies the font to be used. The font is specified using the X-windows convention. The `xfontsel` utility can be used to help select the font string.

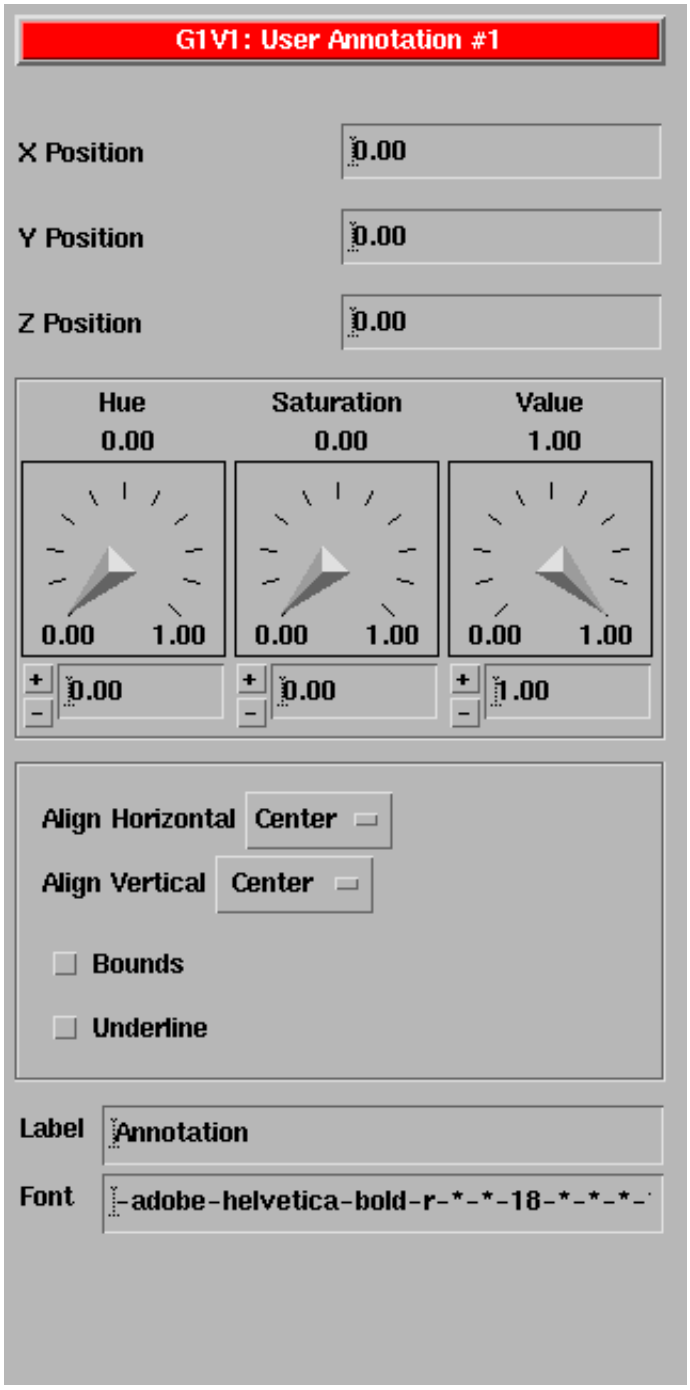


Figure 4.40 Control panel for “User Annotations”

4.6.6 SNL Logo

This visualization method displays the Sandia thunderbird logo in the active viewer. The position of the logo is controlled by the *X*, *Y*, and *Z Position* typein widgets. Valid values for these coordinates are +/- 1. The color of the text is controlled by the HSV color model dial widgets. The size of the logo is controlled with the *Scale* slider widget.

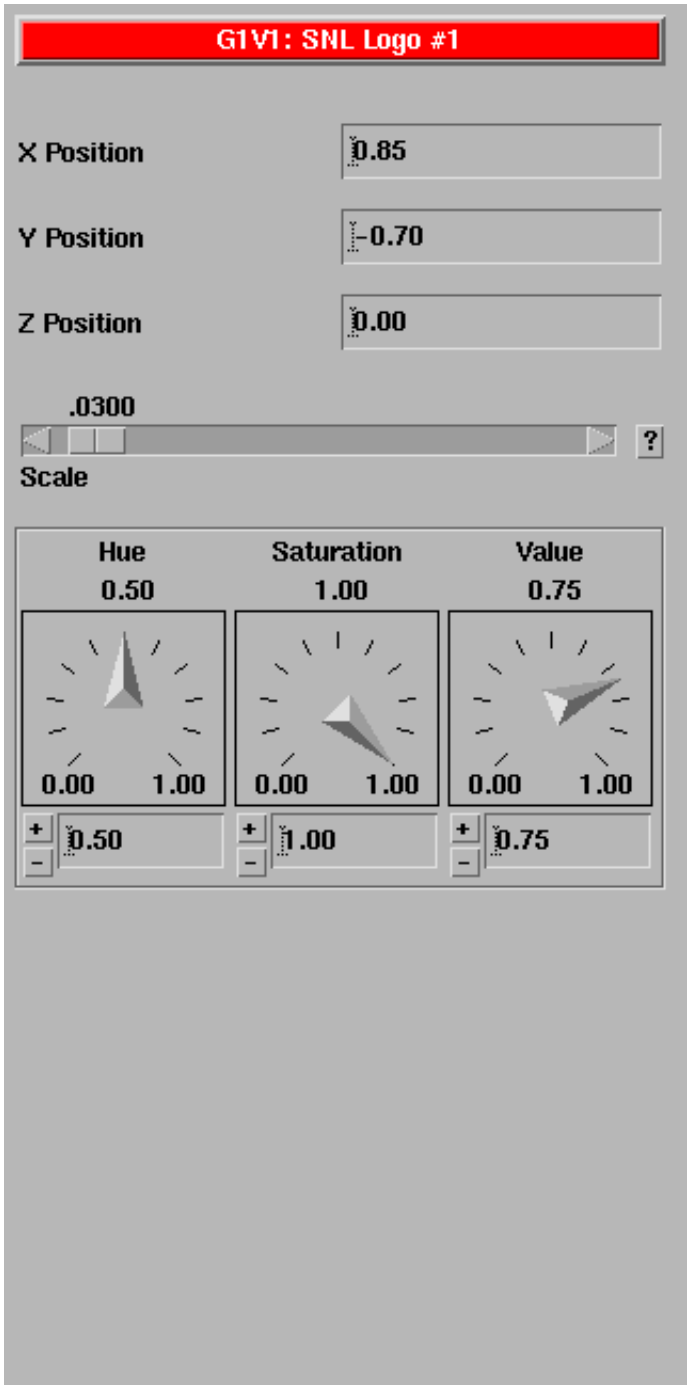


Figure 4.41 Control panel for “SNL Logo”

4.7 Misc Editors

4.7.1 Datamap Legend Editor

This is the popup window used to control the color map legend that is available in some of the visualization methods. It controls the look and visibility of the legend as it appears in the viewers. The *X,Y Min/Max* typein widgets are used to define the four corners of the legend. The *Z Value* typein widget controls the z position (depth) of the legend. The *Z Value* is used to prevent 3D models from obscuring the legend. The *Align Horizontal* and *Align Vertical* controls the alignment of the text labels. The various toggle widgets below the alignment popup menus control other features of the text labels. The *Color* popup menu controls the color of the text labels, tick marks, and legend outline. Only two colors are available, black or white. The *Title Vertical/Horizontal Offset* slider widget controls the offset distance of the variable title in relation to the legend. The *Font* typein widget specifies the font to be used for the title. The *Intervals* popup menu selects the number of intervals on the legend to be labeled. The *Interval Labels Offset* slider widget controls the offset distance of the text labels in relation to the legend. The *Format* typein widget controls the format of the labels. C style formatting convention is used here. The *Font* typein widget specifies the font to be used for the labels. The *Vertical* and *Horizontal* radio button widgets control the orientation of the legend in the viewer. The *Outline* toggle widget controls whether or not an outline is drawn around the legend colorbar. The *Ticks* toggle widget controls whether or not tick marks are drawn across the legend colorbar at the labeled intervals. The *Labels* toggle widget controls whether or not text labels are shown. The *Visible* toggle widget controls whether or not the legend is visible in the viewer. The *Dismiss* button closes this popup window.

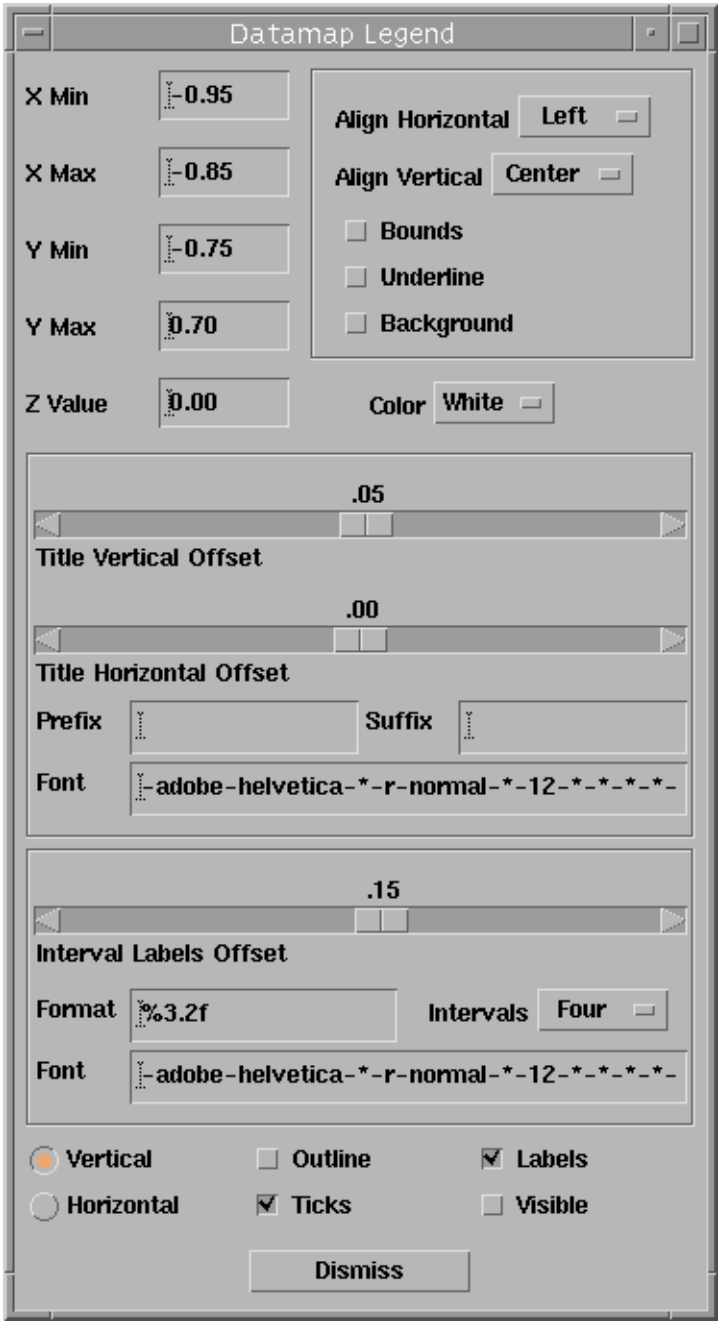
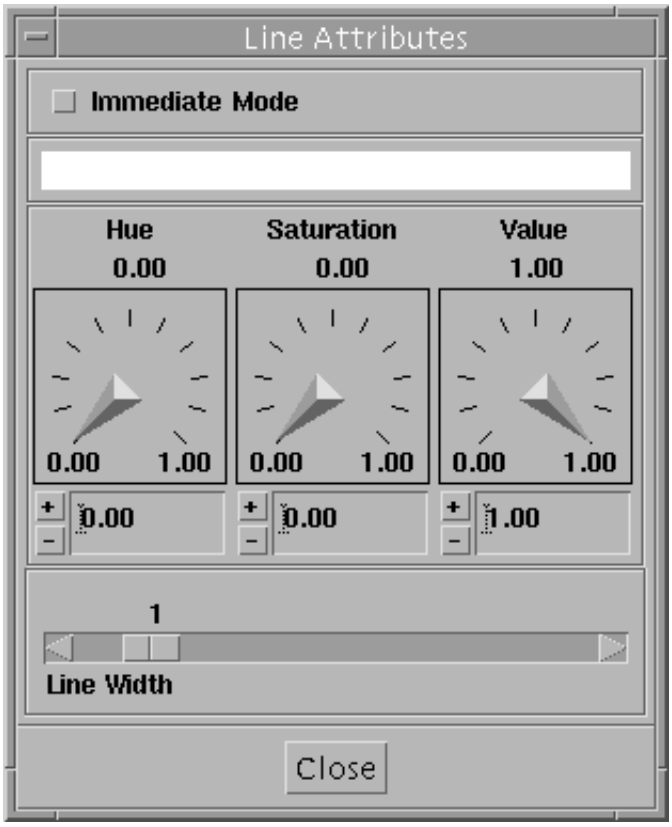
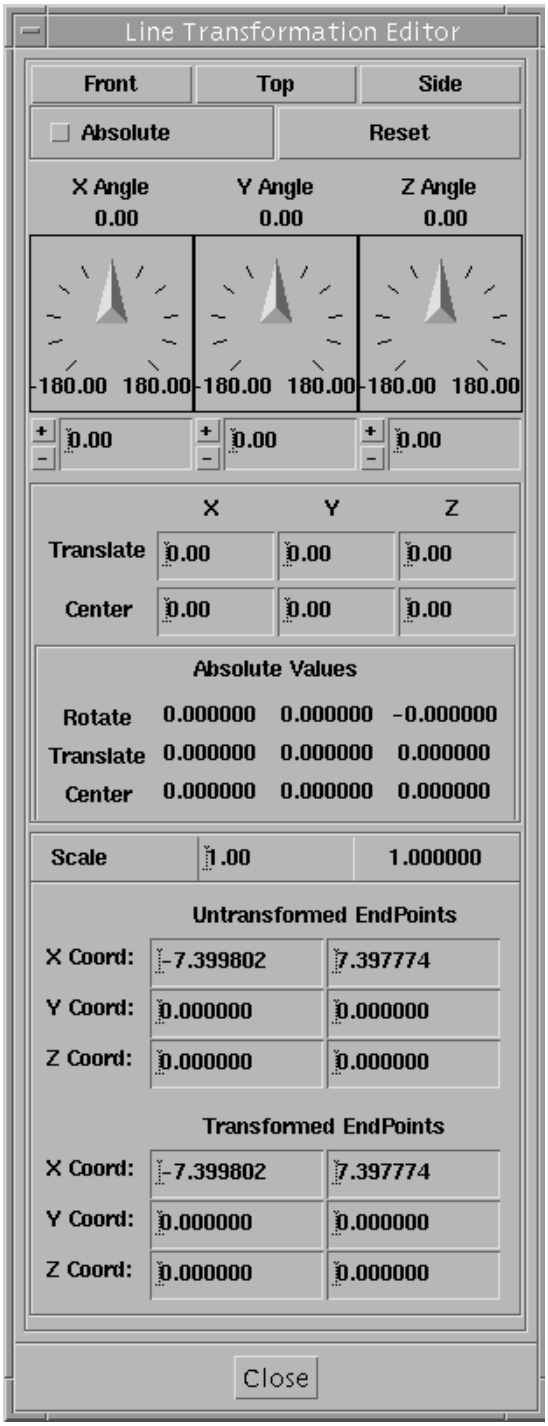


Figure 4.42 Datamap Legend Editor

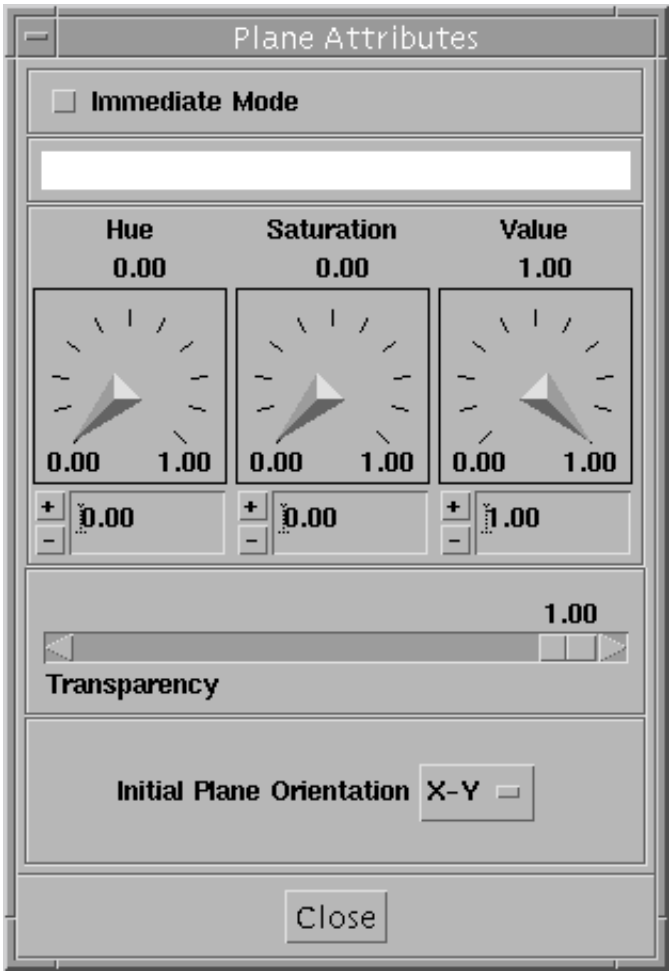
4.7.2 Line Attributes Editor



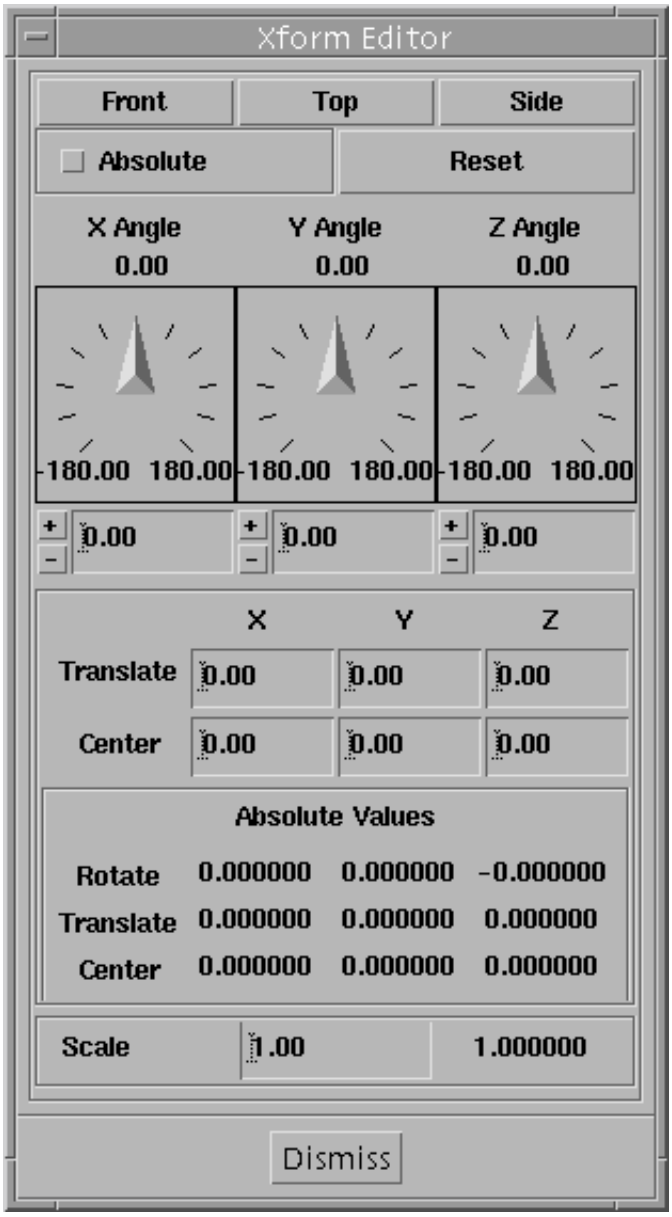
4.7.3 Line Probe Xform Editor



4.7.4 Plane Attributes Editor

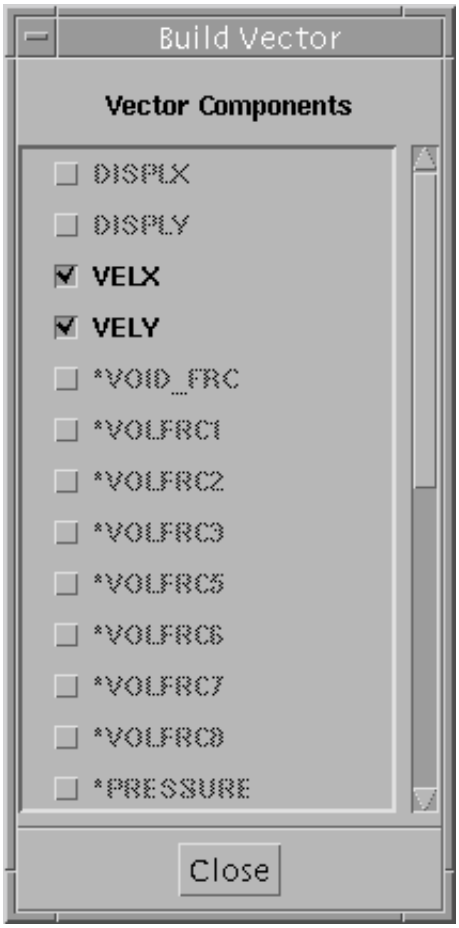


4.7.5 Plane Probe Xform Editor



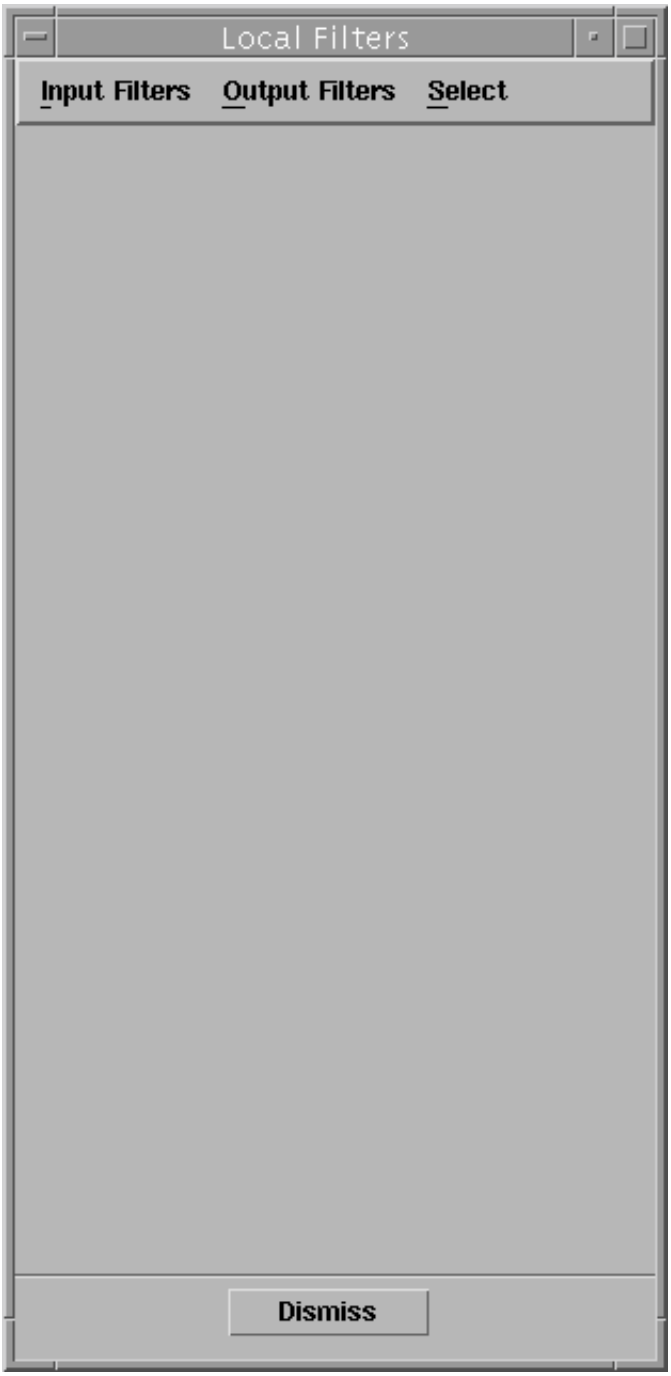
4.7.6 Vector Construction Editor

This is the popup window used for vector construction. It is used to define a vector from scalar components for the vector data visualization methods *(e.g. Hedge Hog, Streamlines, and Particle Advection)*. For 2D datasets, two vector components (x and y) need to be selected and for 3D datasets, three vector components (x, y, and z) need to be selected. The data components are mapped to x, y, and z vector components in the order that they are selected.



5. Global and Local Filters

Global filters are accessed by using the *Filters* pulldown menu in the main control panel. Each Data Input Group has its own set of global filters associated with it. Local filters for each visualization method (for those that support them) are accessed by clicking on the red title bar. Each viz method can support a different set of local filters.



5.1 Global and Local Input FIlters

5.1.1 Mesh Displacements

This filter is used to select which scalar components make up each component of a displacement vector field. The displacement vectors are then applied to the mesh's nodal coordinates to define a new mesh. The displacement vector field can be scaled with the *scale* slider widget.

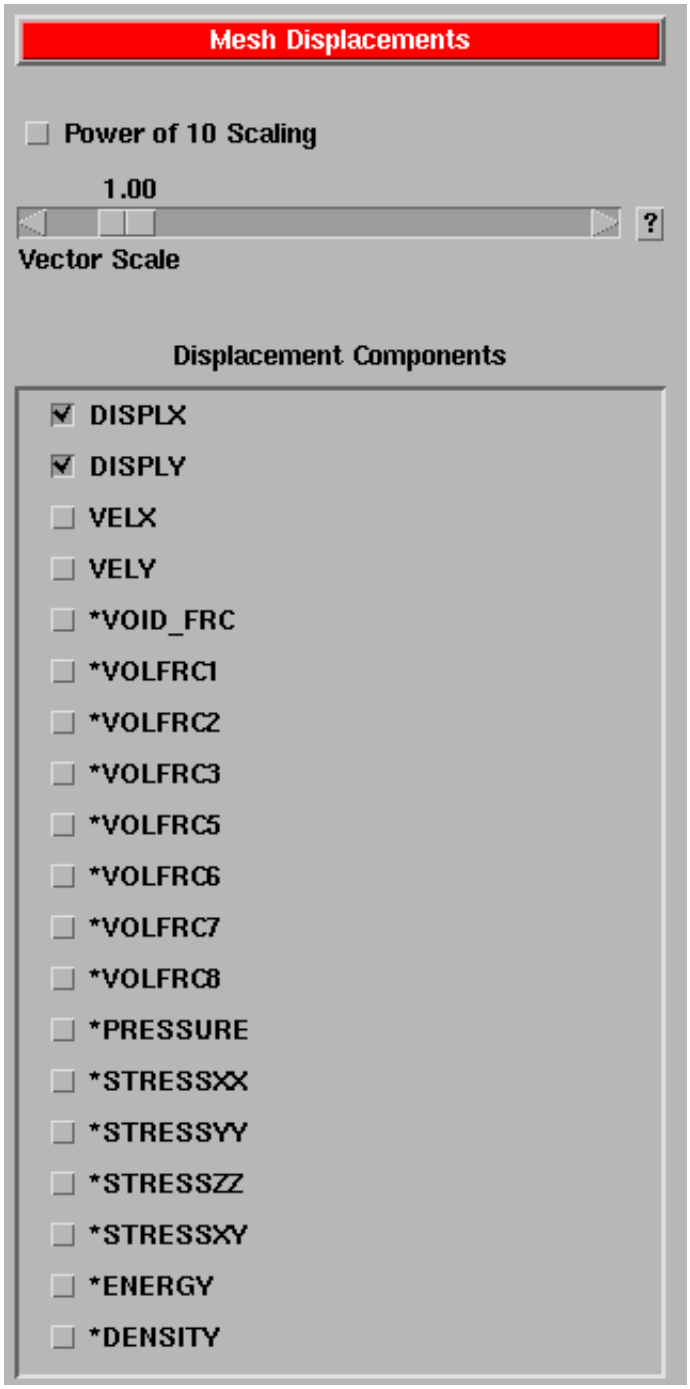


Figure 5.1 Control Panel for Mesh Displacements

5.1.2 Select Element Blocks

This filter is used to select which element blocks in the model are to be passed along in a new mesh to subsequent operations.

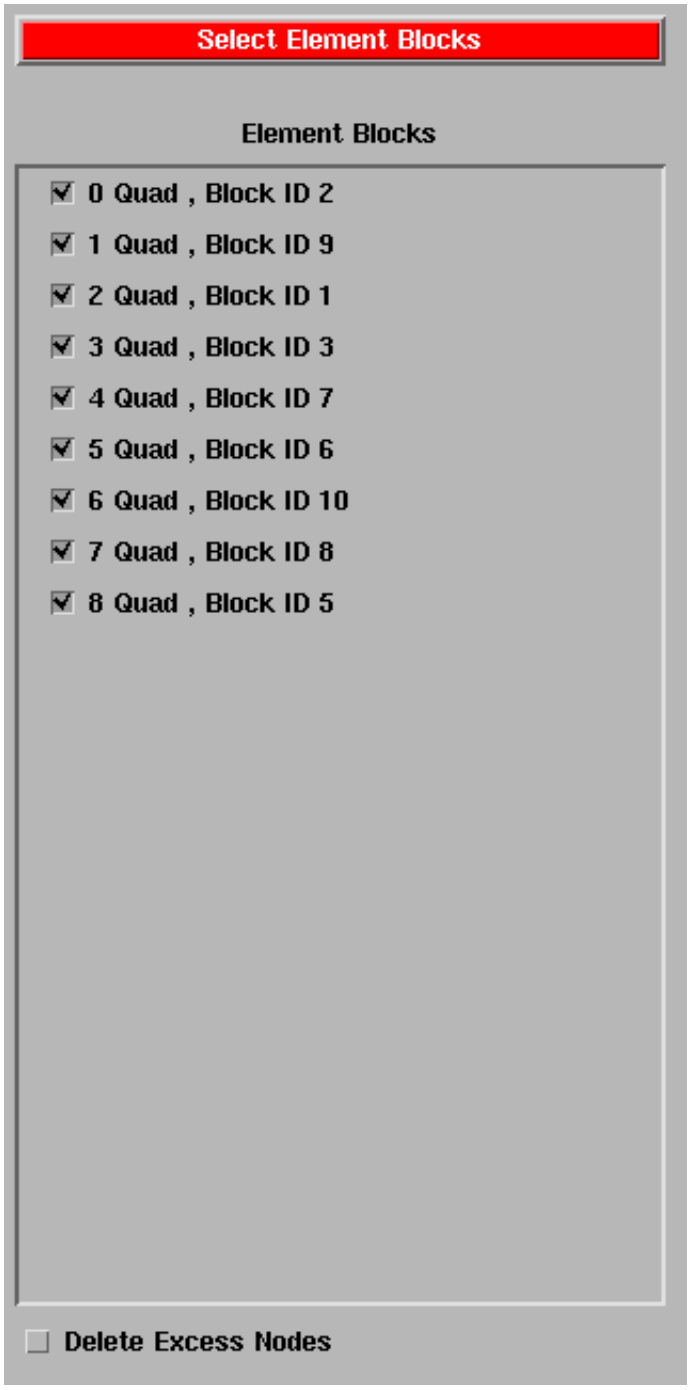


Figure 5.2 Control Panel for Select Element Blocks

5.1.3 Mirror Model

This filter is used to mirror the model about the x-, y-, or z-axis.

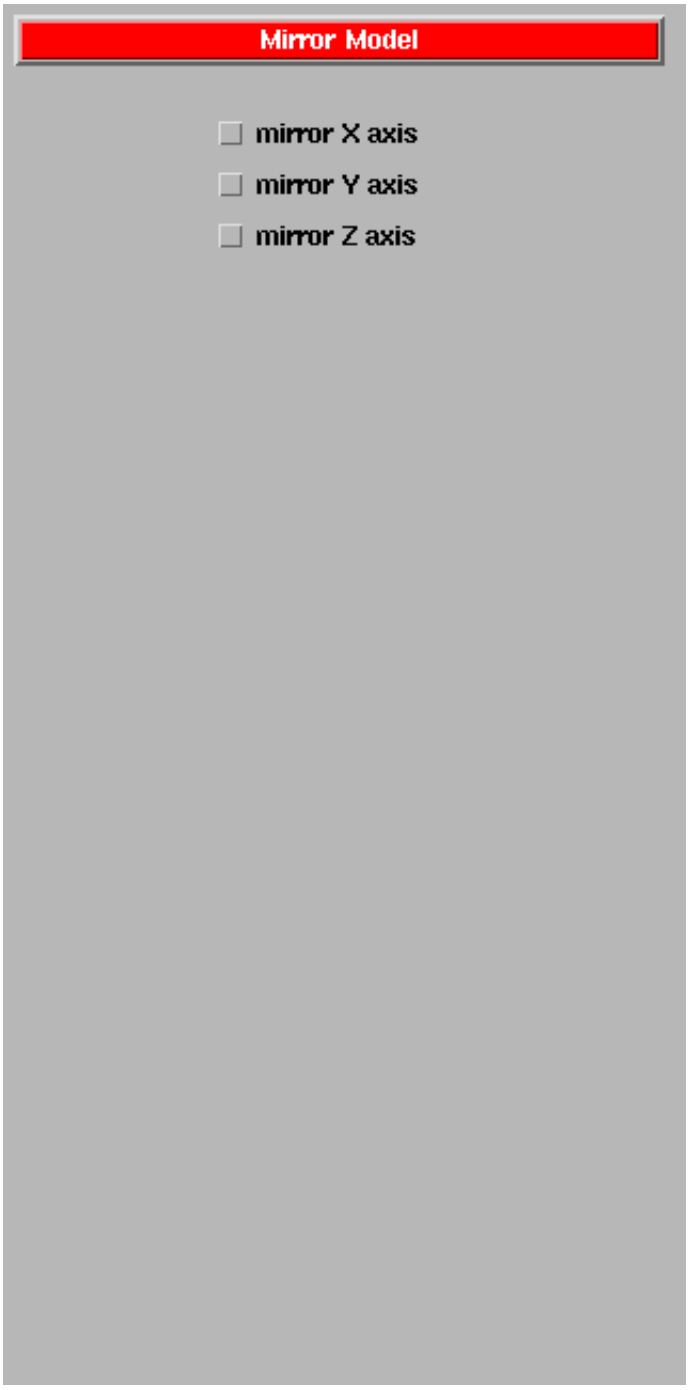


Figure 5.3 Control Panel for Mirror Model

5.1.4 Threshold Cell Data

This filter removes data that falls below and/or above a min/max threshold value. The *Threshold Component* selection box widget is used to set which data components is used to threshold the model. The *Min Value* and *Max Value* typein widgets are used to set the min/max threshold values. The *Below* toggle widget controls thresholding of data below *Min Value*. The *Above* toggle widget controls thresholding of data above *Max Value*.

Figure 5.4 Control Panel for Threshold Cell Data

5.1.5 Threshold Node Data

This filter removes data that falls below and/or above a min/max threshold value. The *Threshold Component* selection box widget is used to set which data components is used to threshold the model. The *Min Value* and *Max Value* typein widgets are used to set the min/max threshold values. The *Below* toggle widget controls thresholding of data below *Min Value*. The *Above* toggle widget controls thresholding of data above *Max Value*.

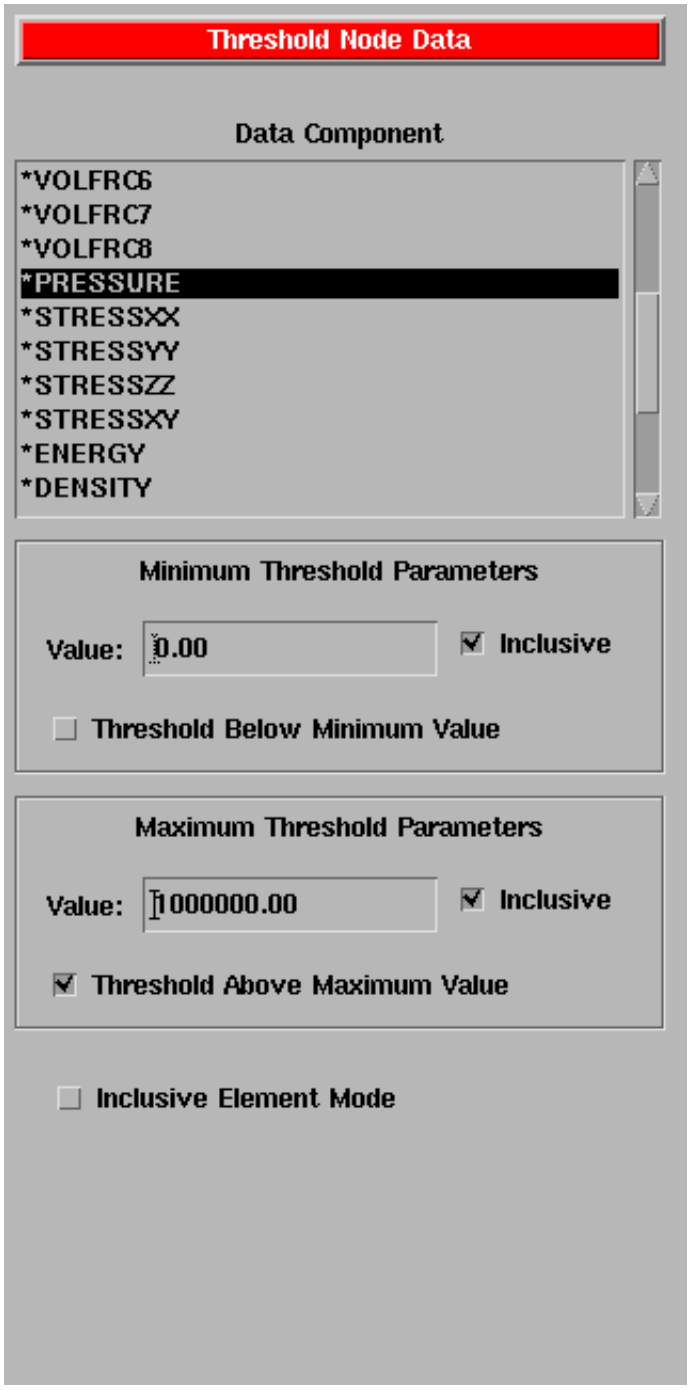


Figure 5.5 Control Panel for Threshold Node Data

5.1.6 Extract Node Data Component

This filter extracts a single scalar component and sets its minimum and maximum values. Setting the minimum and maximum values this way locks in the data-to-color mapping. The *Min Value* and *Max Value* typein widgets are used to set the minimum and maximum values. The *Clamp below* toggle switch is used to have the data field's minimum value overridden with the *Min Value*. The *Clamp above* toggle switch is used to have the data field's maximum value overridden with the *Max Value*. The *Reset Min and Max Values* button widget will reset the *Min Value* and *Max Value* to the data field's minimum and maximum values respectively and turn off the *Clamp below* and *Clamp above* toggle widgets.

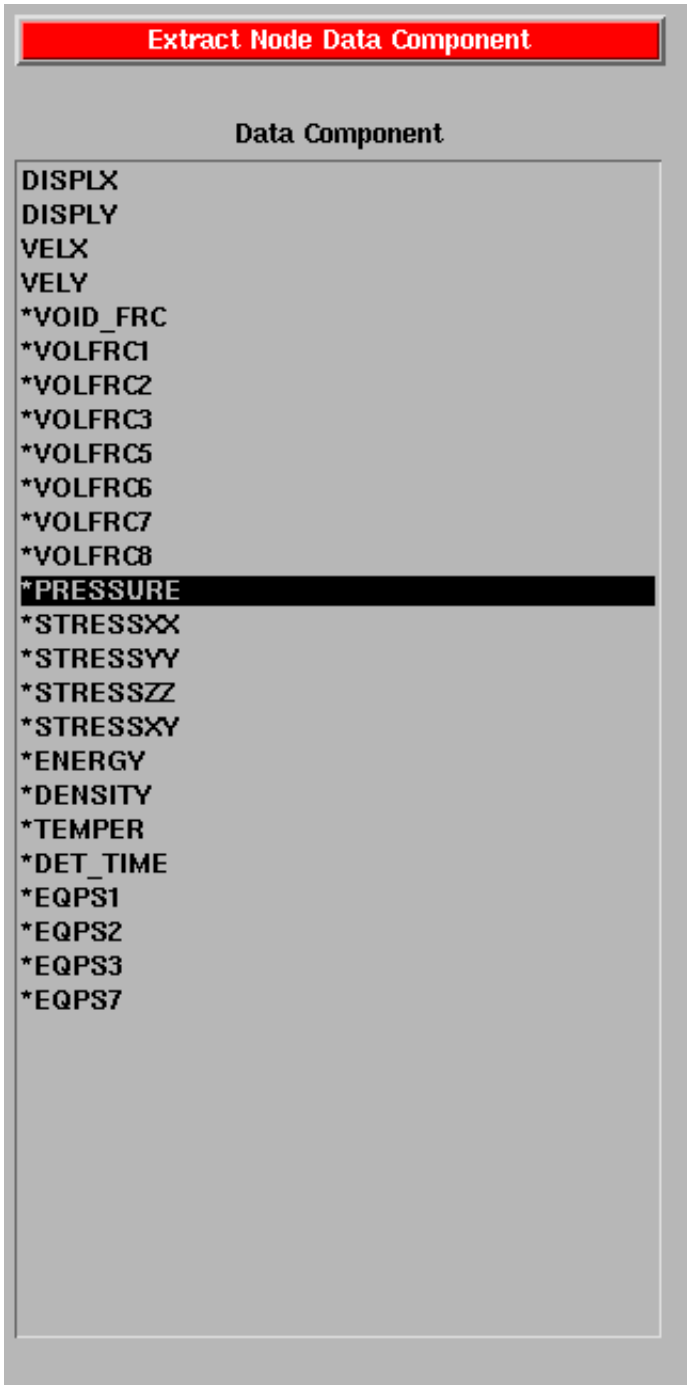


Figure 5.6 Control Panel for Extract Node Data Component

5.1.7 Clamp Node Data Range

This filter extracts a single scalar component and sets its minimum and maximum values. Setting the minimum and maximum values this way locks in the data-to-color mapping. The *Min Value* and *Max Value* typein widgets are used to set the minimum and maximum values. The *Clamp below* toggle switch is used to have the data field's minimum value overridden with the *Min Value*. The *Clamp above* toggle switch is used to have the data field's maximum value overridden with the *Max Value*. The *Reset Min and Max Values* button widget will reset the *Min Value* and *Max Value* to the data field's minimum and maximum values respectively and turn off the *Clamp below* and *Clamp above* toggle widgets.

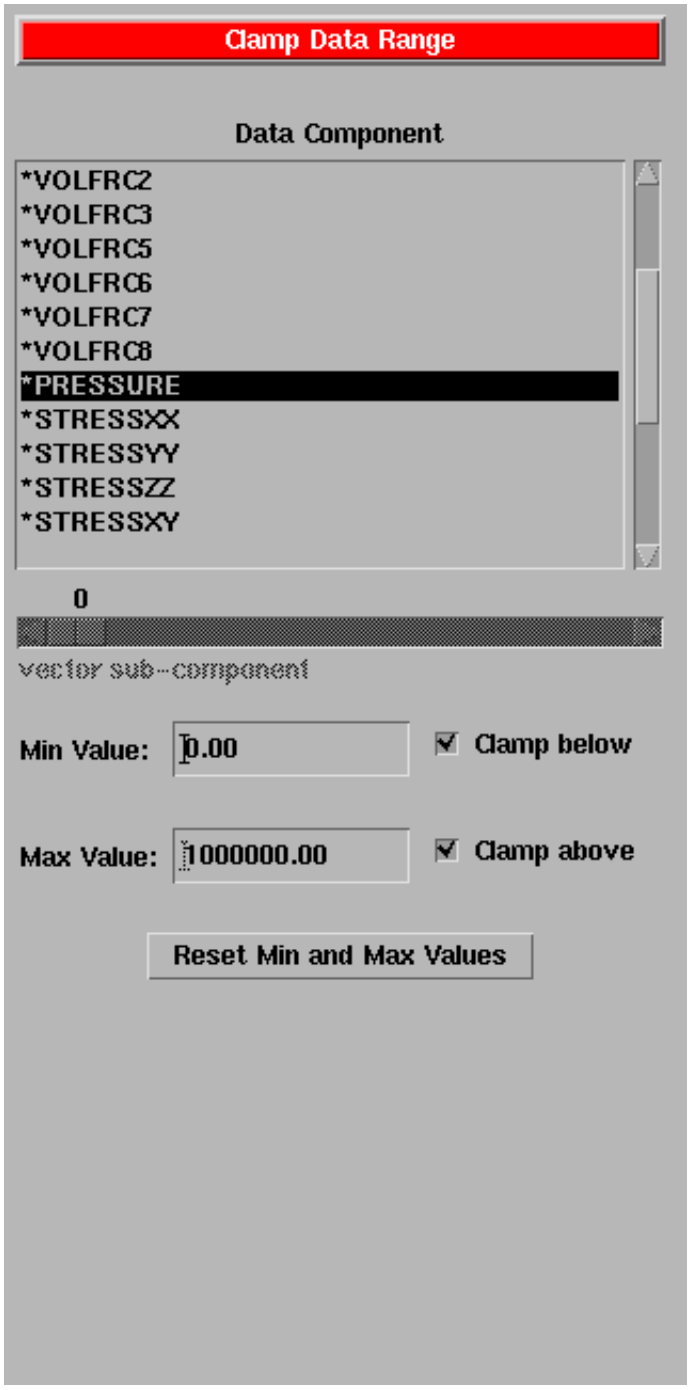
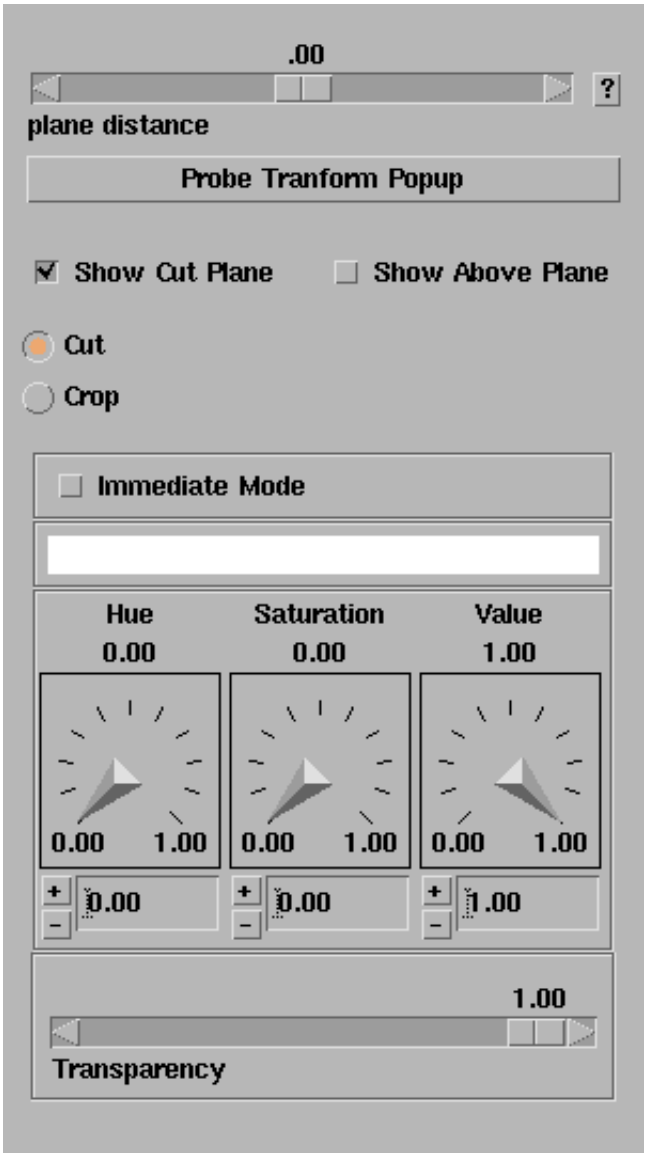


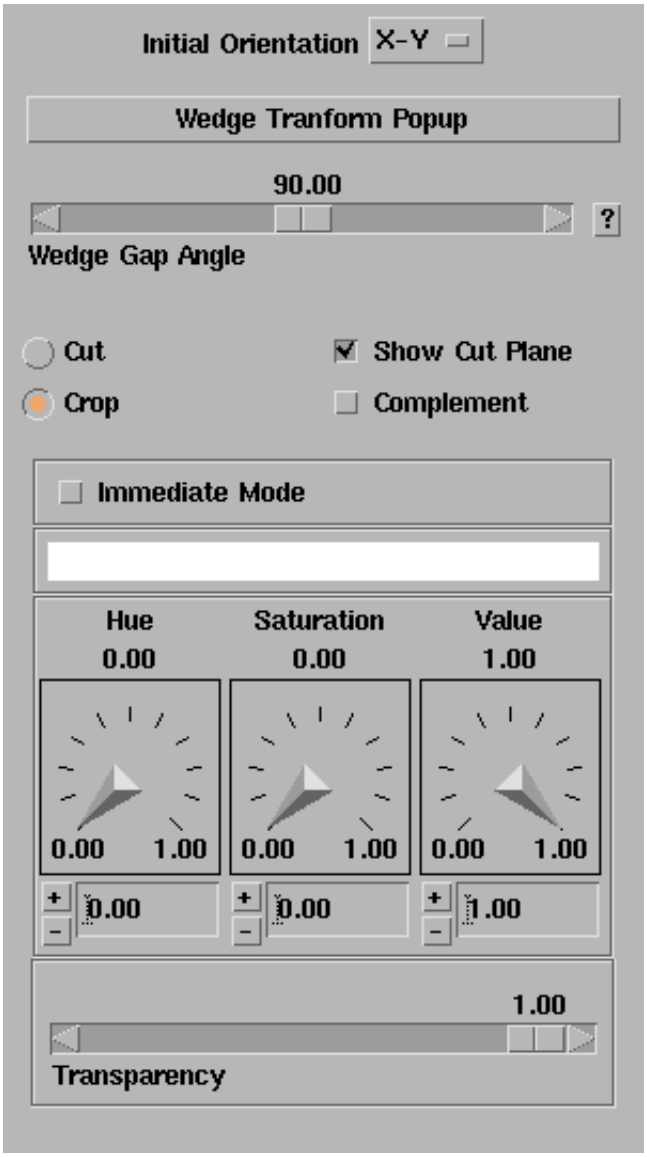
Figure 5.7 Control Panel for Clamp Node Data Range

5.2 Local Output Filters

5.2.1 Plane Cut/Crop

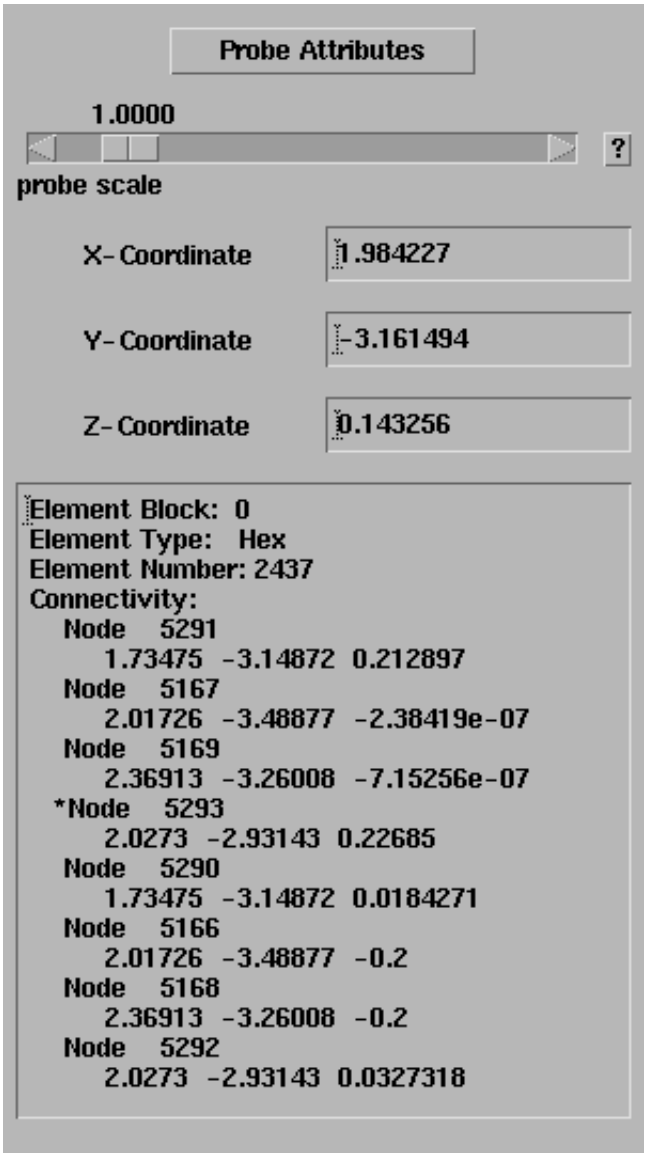


5.2.2 Wedge Cut/Crop



5.3 Miscellaneous Local Filters

5.3.1 Mesh Query



6. Viewer

The MUSTAFA viewer is shown in Figure 6.1. The window's titlebar shows the viewer number. The *Window* pulldown menu provied access to various types of output and window sizes. Current output types are; (1) rasterfile, (2) printfile (postscript), (3) VRML, (4) image sequence, or (5) mpeg-1. The *Axes/Grid* pulldown menu provides access to model axes and grids. These include; (1) *Relative System Axes*, where an axes system is displayed with its origin at the center of the object's transformation matrix, (2) *Absolute System Axes*, where an axes system is displayed with its origin at the object's coordinate frame's origin, and (3) *System Grid*, where a background grid scaled tot he object is displayed, The *Editors* pulldown menus provide access to the viewer controls such as camera position, type of renderer, background color, rendering modes, etc. The viewer also has several of its commonly used functions available in a control area. The *Normalize* button will normalize the current objects to the window's size. The *Center* button will set the geometric center of the viewed object to be the center for all subsequent rotations and zooms. The *Normalize and Center* button will perform both of these function simultaneously. The *Reset* button will reset the viewing transformation to its default state. When an object is first created for the viewer, one usually normalizes and centers it. The *Refresh* button will cause the view to be redrawn. The *Auto* pushbutton will toggle the viewer's automatic refresh on or off. The *Clip* toggle turns the front and back clipping planes on/off. The size of the rendered image is also display along with the currently selected object's name. The popup menu below the SNL logo is used to specify the action of the left-mouse button. The popup menu next to it is used to specify what is transformed by the transformation commands; (1) the objects, (2), the camera, or (3) the lights. The *Current Object* button is used to display a list of all the objects from which one can be selected as the current object.

By default, the following transformation functions are controlled by mouse operations:

1. Scale <LeftMouseButton>: pressing this mouse button and dragging the mouse to the left will zoom out while dragging the mouse to the right will zoom in.
2. Rotate <MiddleMouseButton>: pressing this mouse button and dragging the mouse will rotate the object using a virtual trackball model.
3. Translate RightMouseButton>: pressing this mouse button and dragging the mouse will translate the object according to the mouse movement.
4. Zoom-to-Box <Shift><LeftMouseButton>: pressing this mouse button and dragging the mouse to create a box will cause the viewer to be zoomed to this box when the mouse button is released. With the software renderer, the box can be seen interactively. The center of the box is then the center for all future zooms and rotations.
5. Roam <Cntrl><LeftMouseButton>: pressing on this mouse button while the cursor is on part of the object will cause that point on the object to be translated to center of the viewer. That point will also be the center of any future rotations and translations.

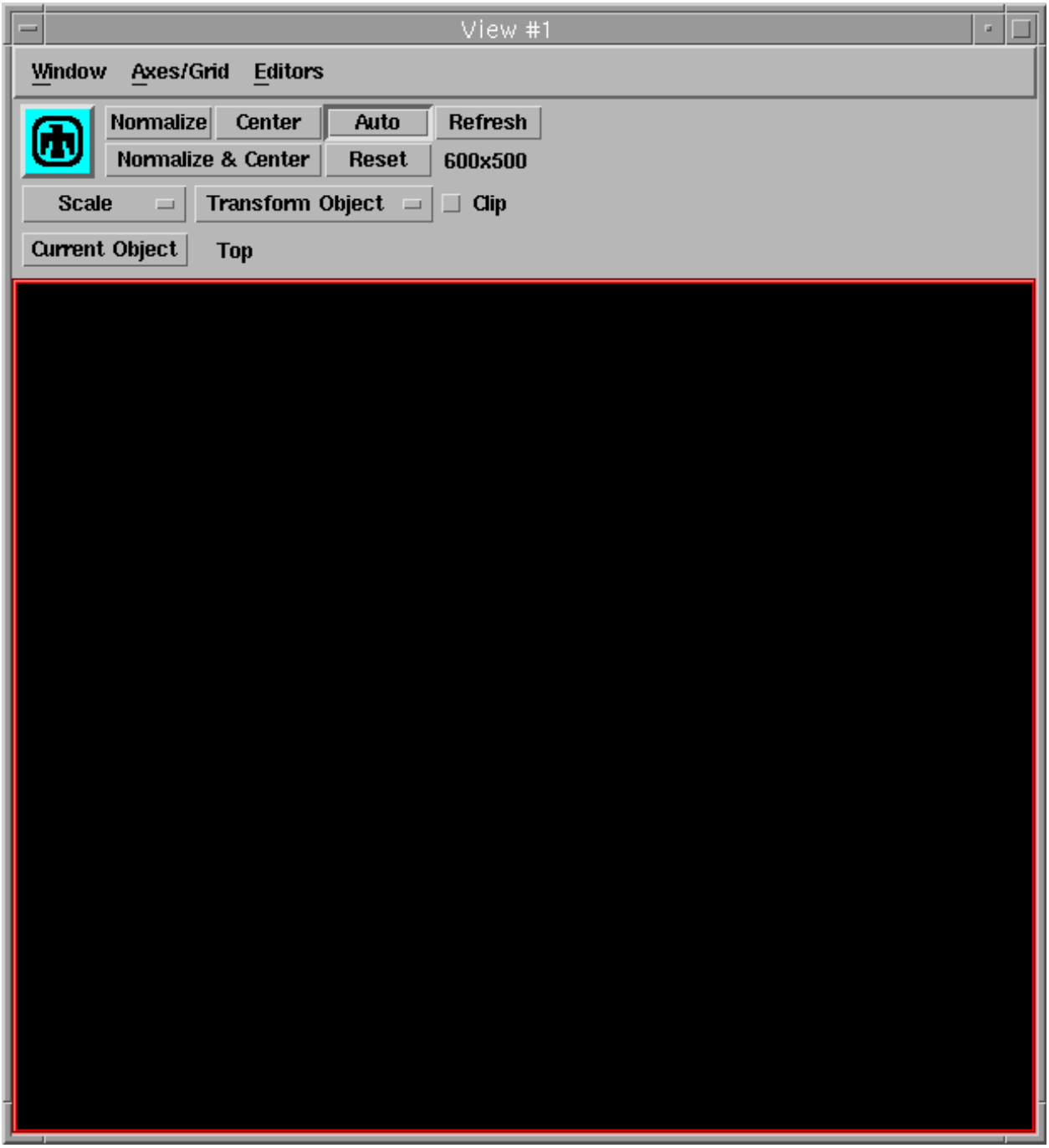


Figure 6.1 Viewer Window

6.1 Image Output

6.1.1 Write Raster File

This module produces a rasterfile of a viewer's contents. The height and width of the rasterfile can be set independently of the viewer's size by using the *Width* and *Height* slider widgets. A sequence number is incremented every time a rasterfile is written to disk and the sequence number can be postpended to the rasterfile name by selecting the *Append file sequence number* toggle widget. The current sequence number can also be changed by typing in a new value into the *Current sequence number* typein widget. The *Flip* toggle widget will vertically flip the image. The *Dynamic* toggle widget will create a new rasterfile every time the active viewer is updated. The use of dynamic rasterfile creation with sequence numbers being appended to the filename is how animation sequences can be saved. The *Swap black & white* toggle widget will swap black and white pixels. This is useful for creating a white background with black lines and text while viewing with the default black background and white text. The *Rasterfile Output* currently supports four (4) image formats: AVS image, TIFF, PPM, and YUV. AVS image format is mainly for compatibility. TIFF is useful for importing into Frame documents. PPM format has a wide variety of image manipulation routines available for it and can be converted into a wide variety of formats. YUV is useful for creating MPEG movie sequences (see *Creating MPEG Movies*). The file name of the rasterfile is specified by typing directly into the *File* typein widget or a Motif file browser can be used by selecting the "..." button. If not in dynamic mode, selecting the *Create Raster File* button widget will create the raster file.

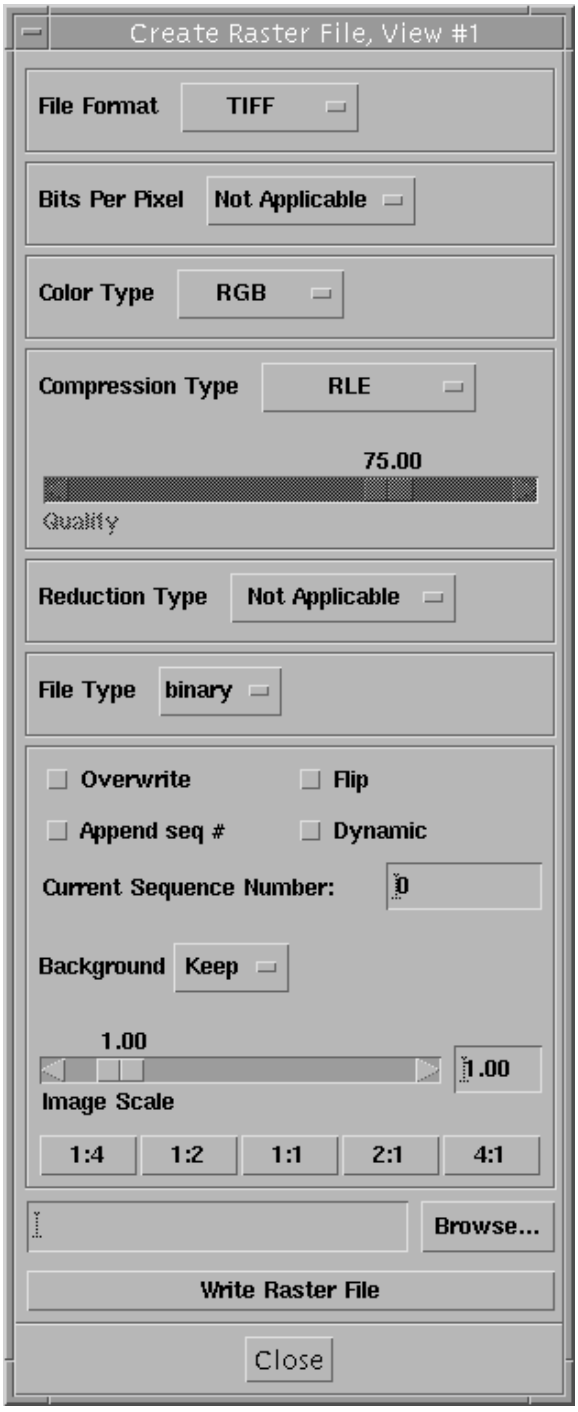


Figure 6.2 Control panel for "Write Rasterfile"

6.1.2 Write Print File

This module produces a printfile of a viewer's contents. The *Format* popup menu is used to specify the file format. Various forms of postscript and CGM are supported. The *Orientation* popup menu is used to select the orientation of the image on the print page. The *Background* popup menu allows for specifying the background color to be black, white, or as is. The *Size* popup menu controls the size of the print page. The *widthMM* and *heightMM* typein widgets are used to specify the image size, in mm, for a user defined page size (i.e. EPS). The *3D Horizontal Resolution* slider widget is used to specify the number of pixels in the X direction of the 3D frame buffer. The typein widget is used to set the output file name. Clicking on the button, *"Browse..."*, to the right of this widget will bring up a standard Motif filebrowser dialog window that can also be used to select the file name. Clicking on the *Write Print File* button widget will cause the print file to be created.

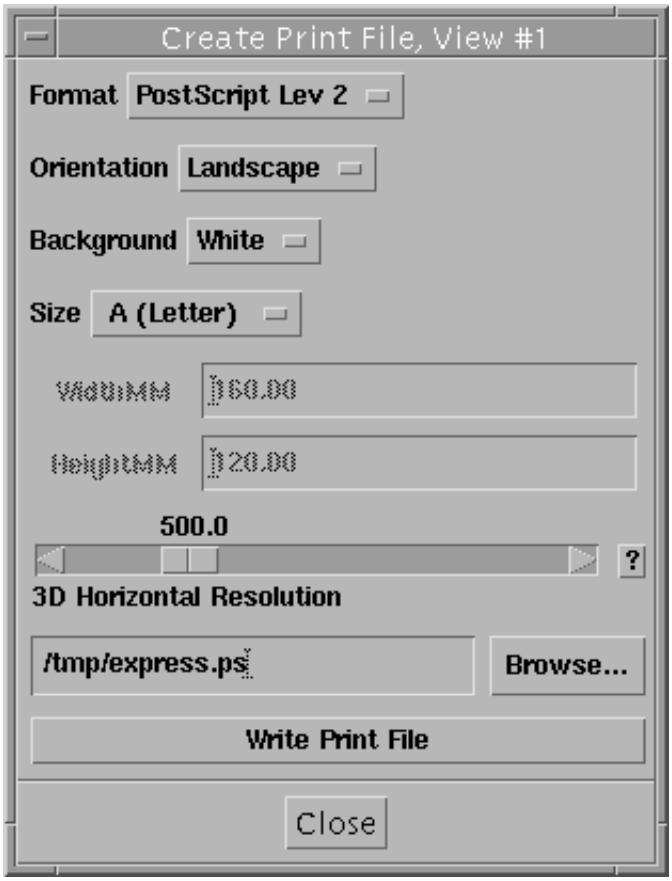


Figure 6.3 Control panel for "Write Printfile"

6.1.3 Write VRML File

This module produces a VRML of a viewer's contents.

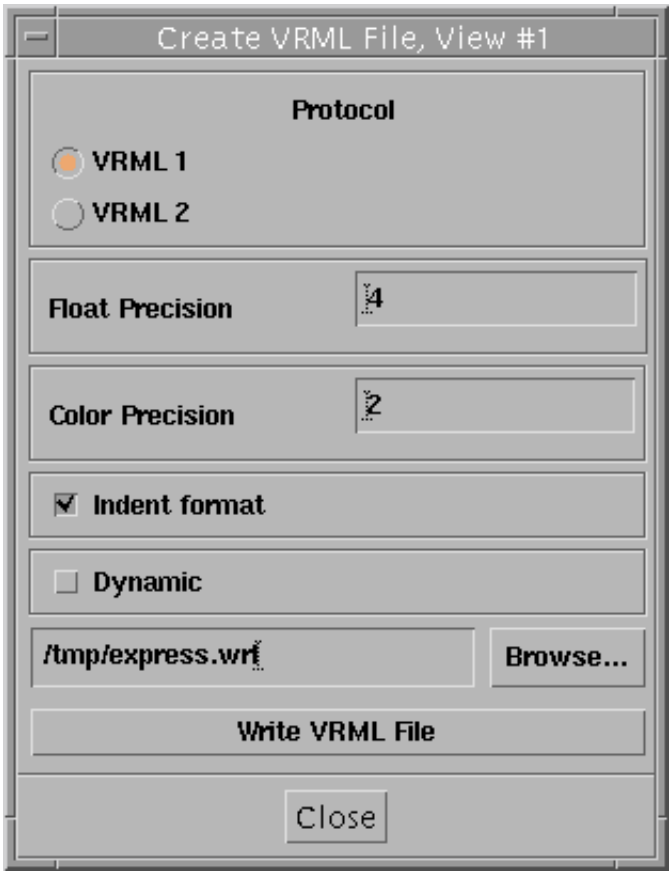


Figure 6.4 Control panel for "Write VRML"

6.1.4 Write Image Sequence

This module produces an image sequence file from a series of a viewer's contents. The *Current Frame* widget displays how many frames have been saved to the sequence file. The *Write Function* radio button turns the sequencing on or off. The output file is not explicitly closed until a new filename is specified. The *Swap Black & White* toggle will casue black and while pixels to be swapped during the save process. The *Background* popup menu allows for specifying the background color to be black, white, or as is. The *Compression* popup menu allows the user the select the type of compression scheme to use while saving the image. The *Image Scale* widgets will allow the image to scaled before being saved. The typein widget is used to set the output file name. Clicking on the button, "*Browse...*", to the right of this widget will bring up a standard Motif filebrowser dialog window that can also be used to select the file name.

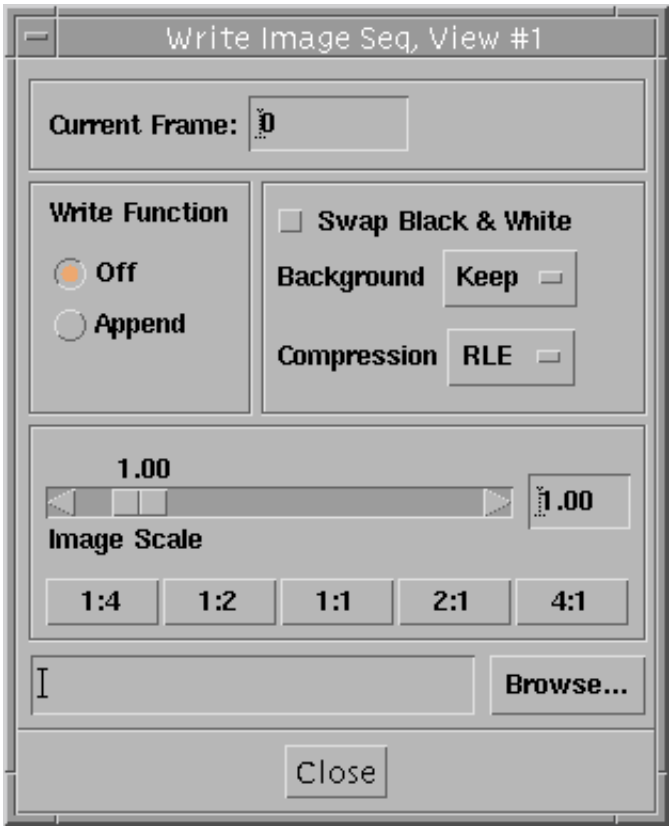


Figure 6.5 Control panel for "Write Image Sequence"

6.1.5 Write MPEG Sequence

This module produces a MPEG-1 video file from a series of a viewer's contents. The *Current Frame* widget displays how many frames have been saved to the mpeg file. The *Write Function* radio button turns the mpeg encoding on or off. The the ecoding is turned off, the output file is automatically closed. The *General* button will bring up an additional popup window (Figure 6.7) for setting the mpeg encoding parameters. The mpeg encoder used in MUSTAFA is the Berkeley encoder (v1.5b) so a description of the parameters can be found in the Berkeley encoder's documentation. The *Image Scale* widgets will allow the image to scaled before being saved. The typein widget is used to set the output file name. Clicking on the button, "*Browse...*", to the right of this widget will bring up a standard Motif file-browser dialog window that can also be used to select the file name. The normal series of operations would be:

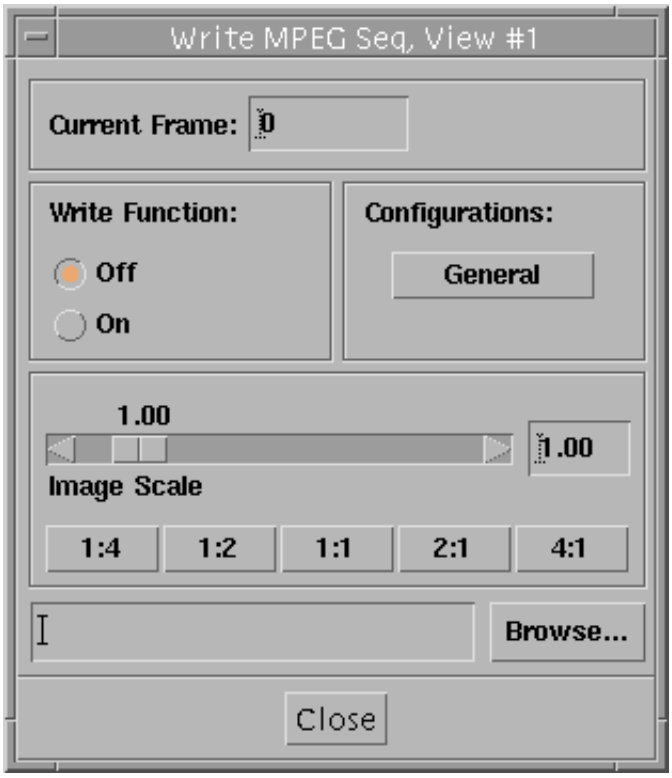


Figure 6.6 Control panel for "Write MPEG Sequence"

1. Set the mpeg encoding options.
2. Set the output file name, the encoding options are copied at this time.
3. Turn the encoding on. The current image is not encoded. To force encoding of the current image, click on the viewer's *Refresh* button. All subsequent images rendered in the viewer will be encoded until the encoding is turned off.

By default, the encoder will encode I-frames only. When encoding a sequence of images of unknown length, the encoder will not properly encode the last frame if it happens to be a B-frame since it depends upon a future I- ot P-frame that will not be encoded. To avoid this problem, only encode I-frames or I-frames and P-frames. The simplest solution is to encode only I-frames. This will not produce as much compression as I-, B-, and P-frame encoding encoding but does make it possible to produce a MPEG video from within MUSTAFA.

6.1.6 Other Methods for Creating MPEG-1 Movies

MPEG-1 movies can also be generated by writing a series of rasterfiles to disk and using any one of a number of software encoders (such as the Stanford encoder, mpeg, ot the Berkeley encoder, mpeg_encode) to encode the indivdual rasterfiles into a MPEG-1 movie. For example, if the *Rasterfile Output* module is used to generate a sequence of TIFF formatted rasterfiles:

```
test.tif.0    test.tif.1    test.tif.2    test.tif.3    test.tif.4
test.tif.5    test.tif.6    test.tif.7    test.tif.8    test.tif.9
```

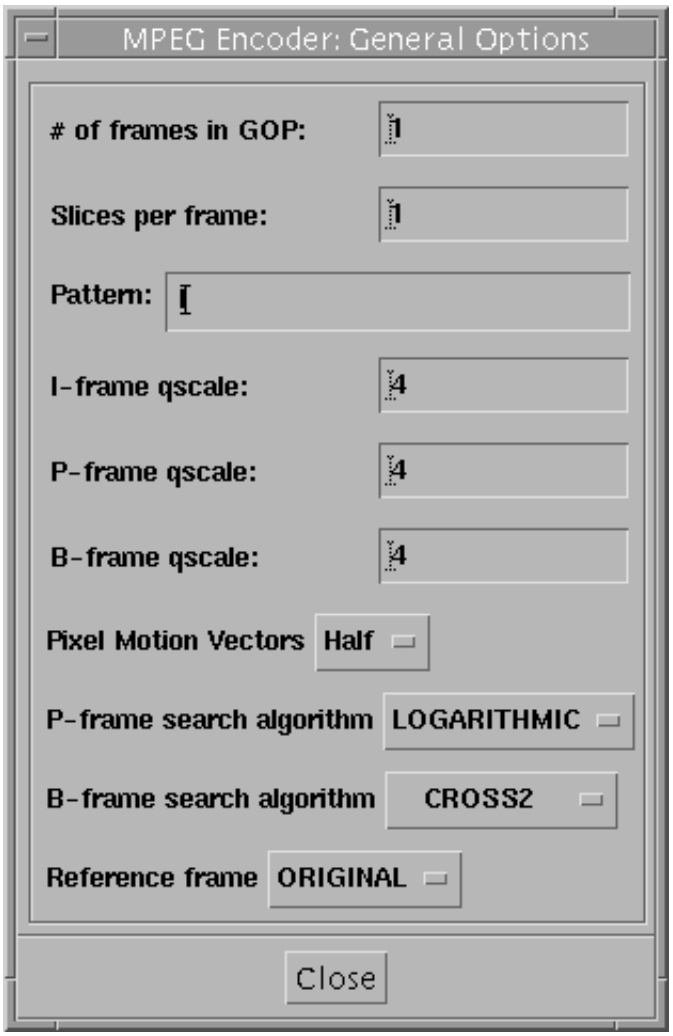


Figure 6.7 Control panel for setting the mpeg-1 encoding parameters

```
test.tif.10    test.tif.11    test.tif.12    test.tif.13    test.tif.14
test.tif.15    test.tif.16    test.tif.17    test.tif.18    test.tif.19
test.tif.20    test.tif.21    test.tif.22    test.tif.23    test.tif.24
test.tif.25    test.tif.26    test.tif.27    test.tif.28    test.tif.29
```

The mpeg-1 encoder from the Portable Video Research Group at Stanford University can be used to encode the image sequence into a mpeg1 file, test.mpg, by running a script, `makemovie 0 29 test`, where makemovie is given as:

```
#! /bin/csh -f
```

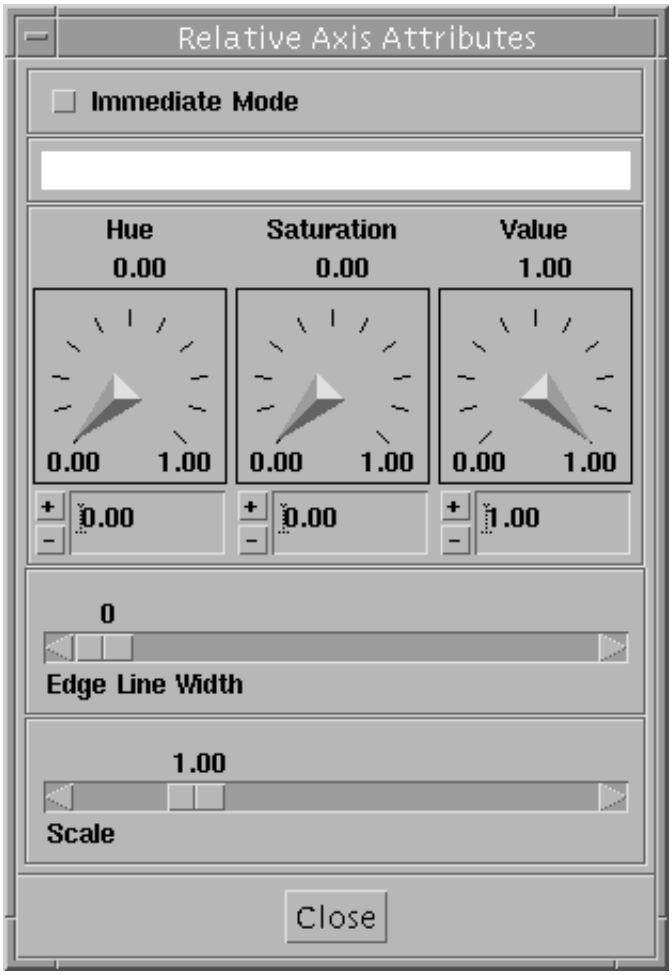
```
set start = $1
set stop = $2
set base = $3
set nframes = $base
@nframes++set size = ‘tiffsize $base.tif.$start‘
set w = size[0]
set h = size[1]
set frame = $start
while ($frame < $nframes) then
     tifftoppm $base.tif.$frame > tmp.ppm
     ppm2cyuv $tmp.ppm $base$frame
     rm tmp.ppm $base.tif.$frame
     @frame++
end
mpeg -PF -a $start -b $stop -h $w -v $h -s $base.mpg $base
rm $base*.Y $base*.U $base.V
```

The mpeg-1 encoder from the University of California Berkeley can also be used to encode the same rasterfiles by using the parameter file:

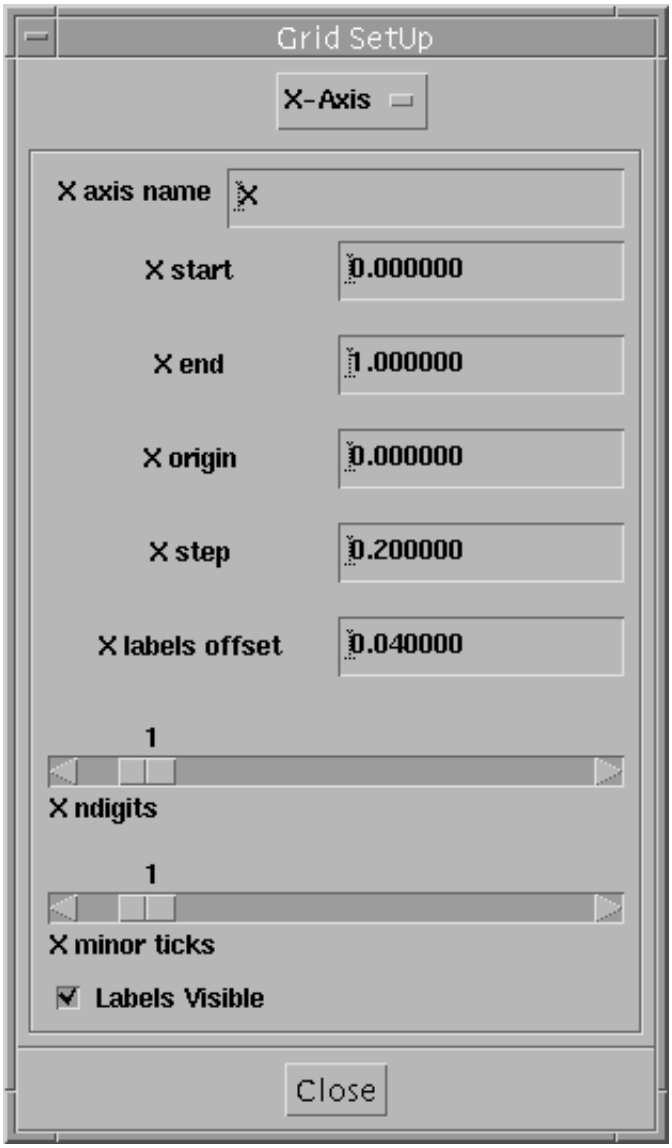
```
# Simple parameter file
OUTPUT                ./test.mpg
GOP_SIZE              15
SLICES_PER_FRAME      1
BASE_FILE_FORMAT      PPM
INPUT_CONVERT         tifftoppm *
INPUT_DIR             .
INPUT
test.tif.* [0-29]
END_INPUT
PATTERN               IBBPBBPBBPBBPBBPBBPBBP
IQSCALE               4
PQSCALE               4
BQSCALE               4
PIXEL                 HALF
RANGE                 10
PSEARCH_ALG           LOGARITHMIC
BSEARCH_ALG           CROSS2
REFERENCE_FRAME       ORIGINAL
```

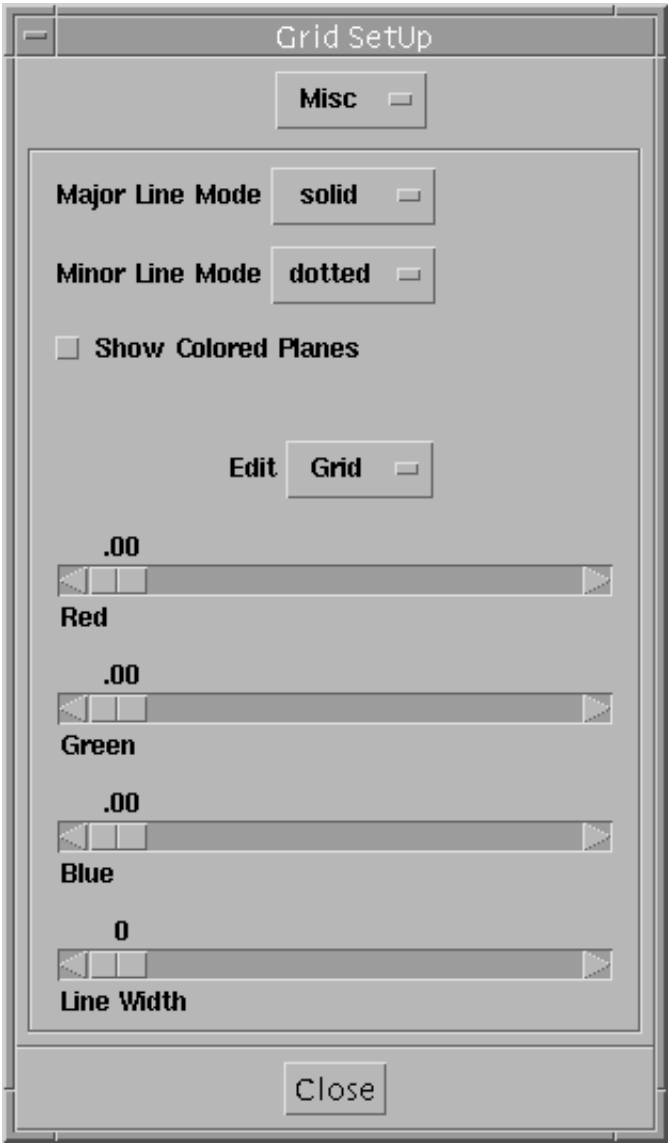
6.2 Axis/Grid Editors

6.2.1 Relative/Absolute Axis Editor



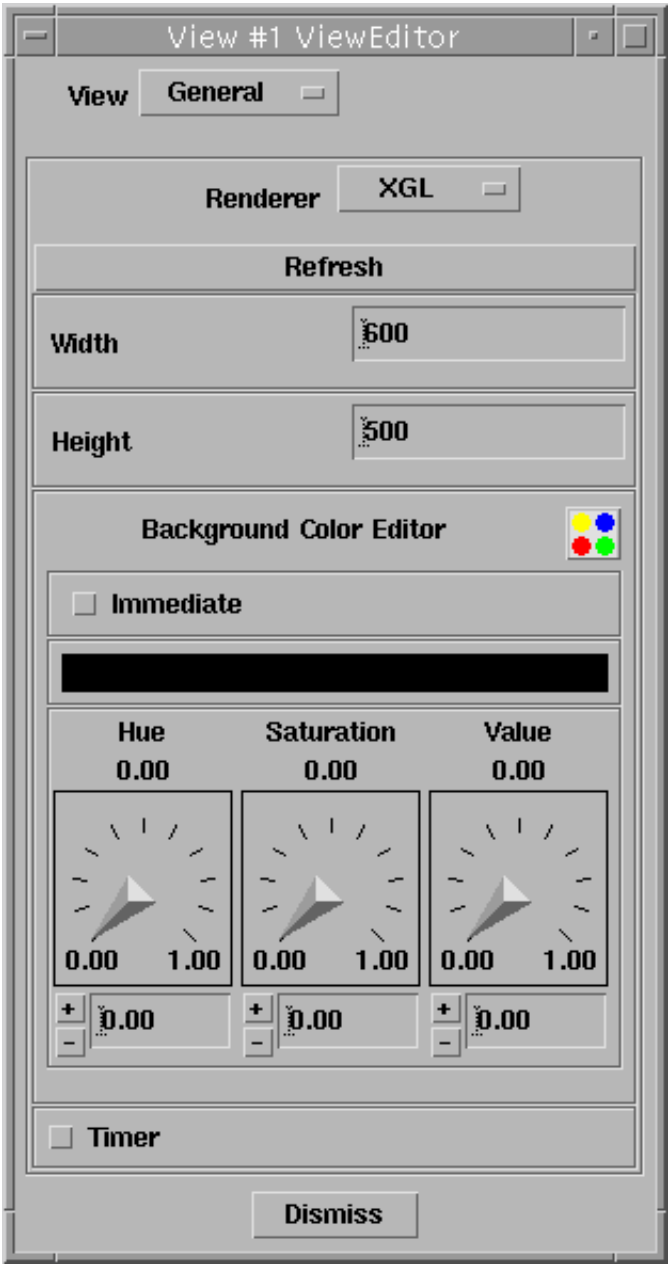
6.2.2 Grid Editor

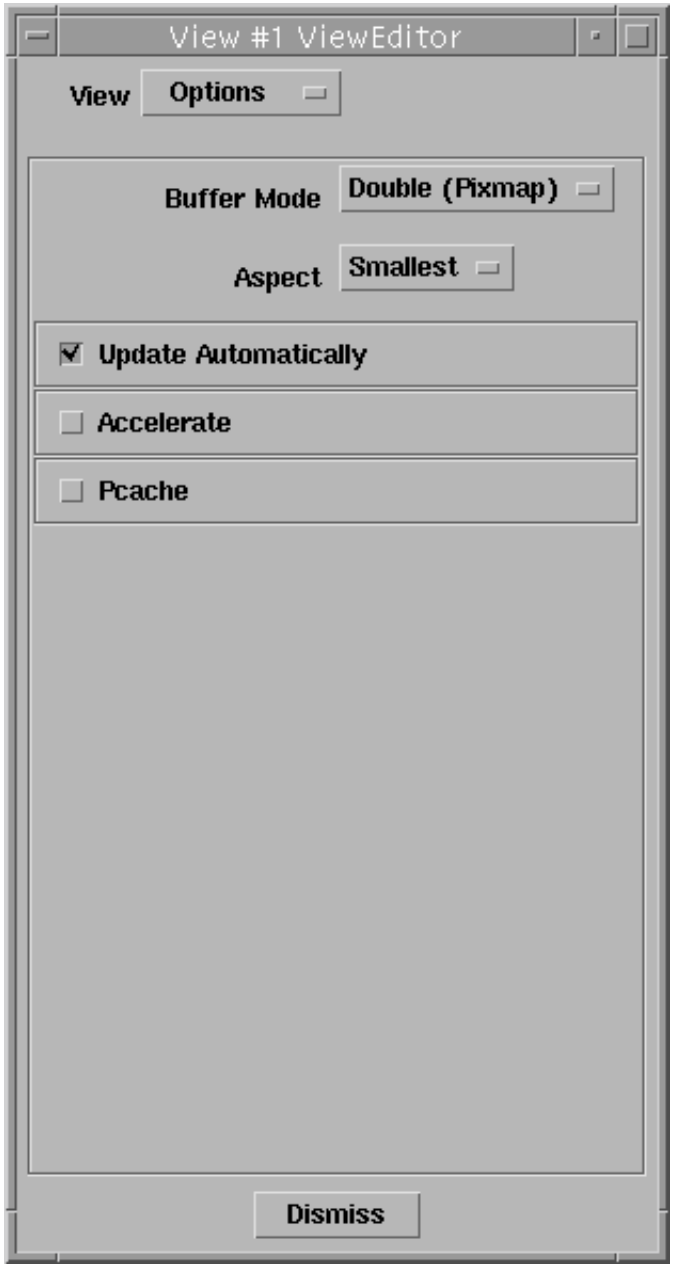



Grid SetUp
Misc
Major Line Mode
solid
Minor Line Mode
dotted
Show Colored Planes
Edit
Grid
.00
Red
.00
Green
.00
Blue
0
Line Width
Close

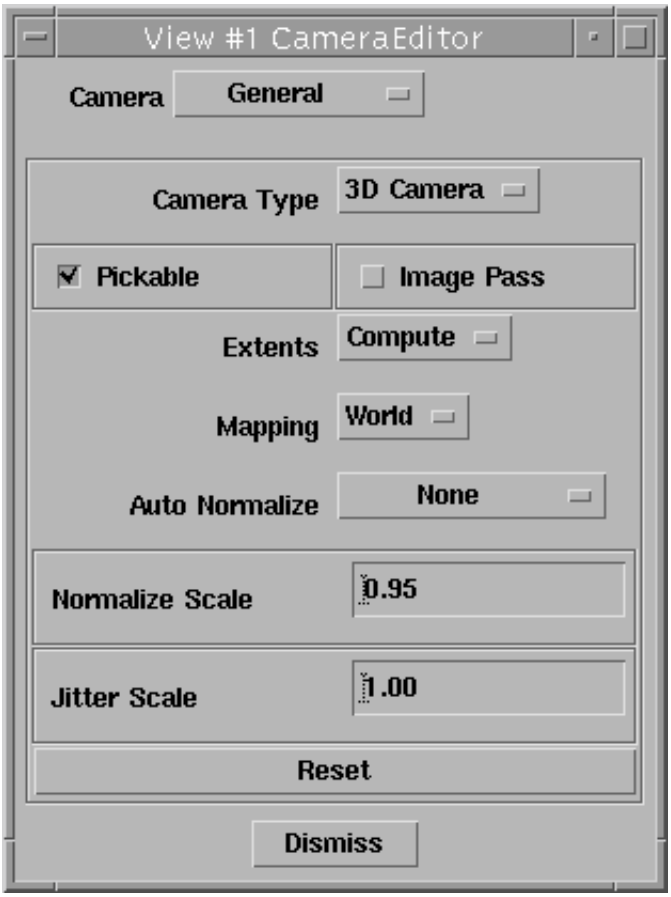
6.3 Graphics Display Editors

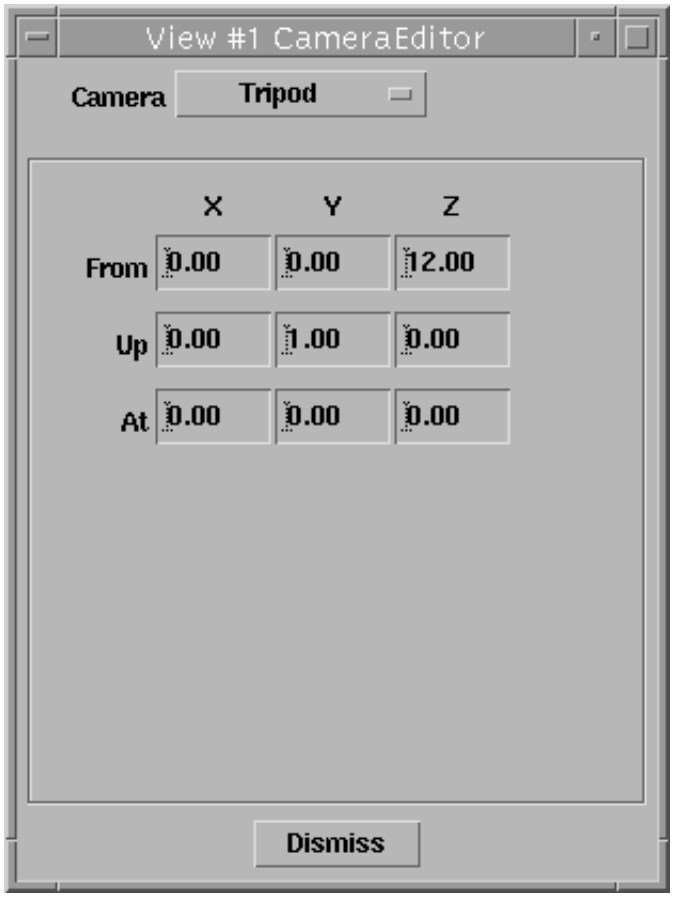
6.3.1 View Editors

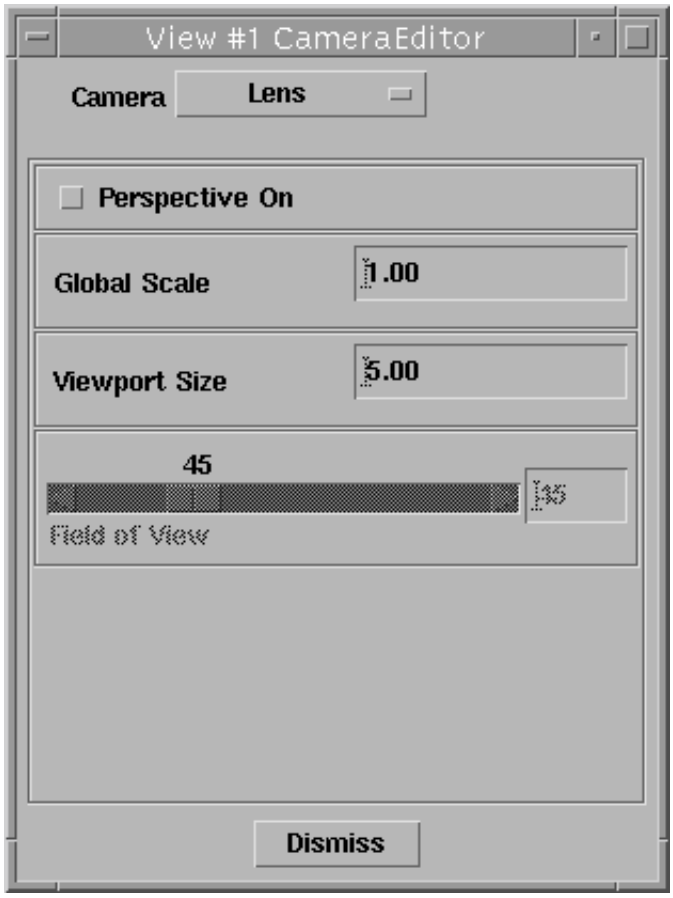

View #1 ViewEditor
View
General
Renderer
XGL
Refresh
Width
600
Height
500
Background Color Editor
Immediate
Hue
Saturation
Value
0.00
0.00
0.00
0.00
1.00
0.00
1.00
0.00
1.00
0.00
0.00
0.00
Timer
Dismiss


View #1 ViewEditor
View
Options
Buffer Mode
Double (Pixmap)
Aspect
Smallest
Update Automatically
Accelerate
Pcache
Dismiss

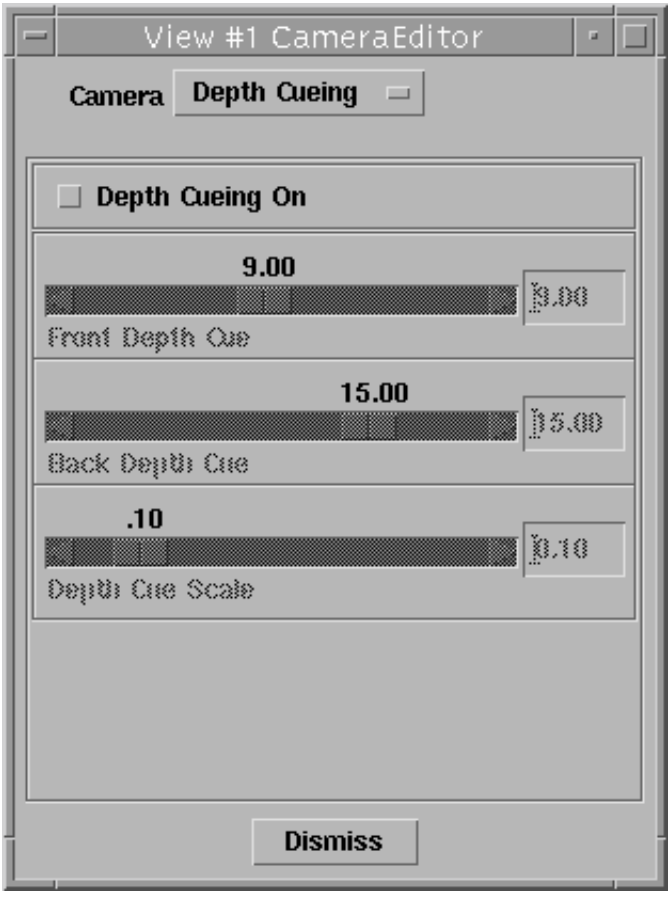
6.3.2 Camera Editors


View #1 CameraEditor
Camera
General
Camera Type
3D Camera
Pickable
Image Pass
Extents
Compute
Mapping
World
Auto Normalize
None
Normalize Scale
0.95
Jitter Scale
1.00
Reset
Dismiss

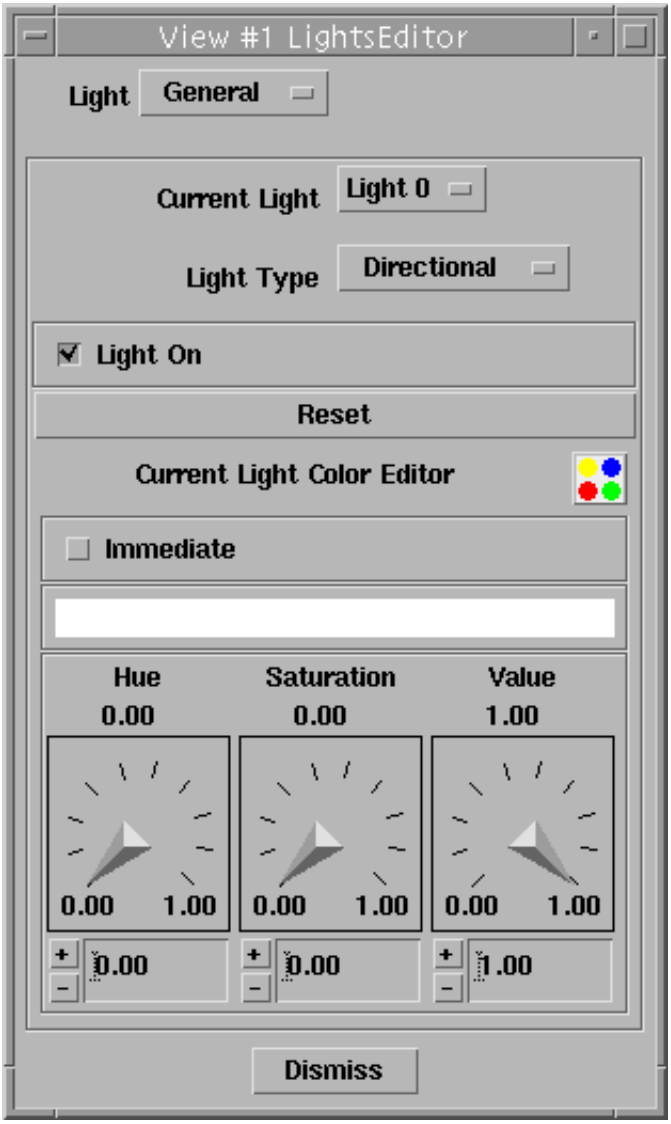

View #1 CameraEditor
Camera
Tripod
X
Y
Z
From
0.00
0.00
12.00
Up
0.00
1.00
0.00
At
0.00
0.00
0.00
Dismiss

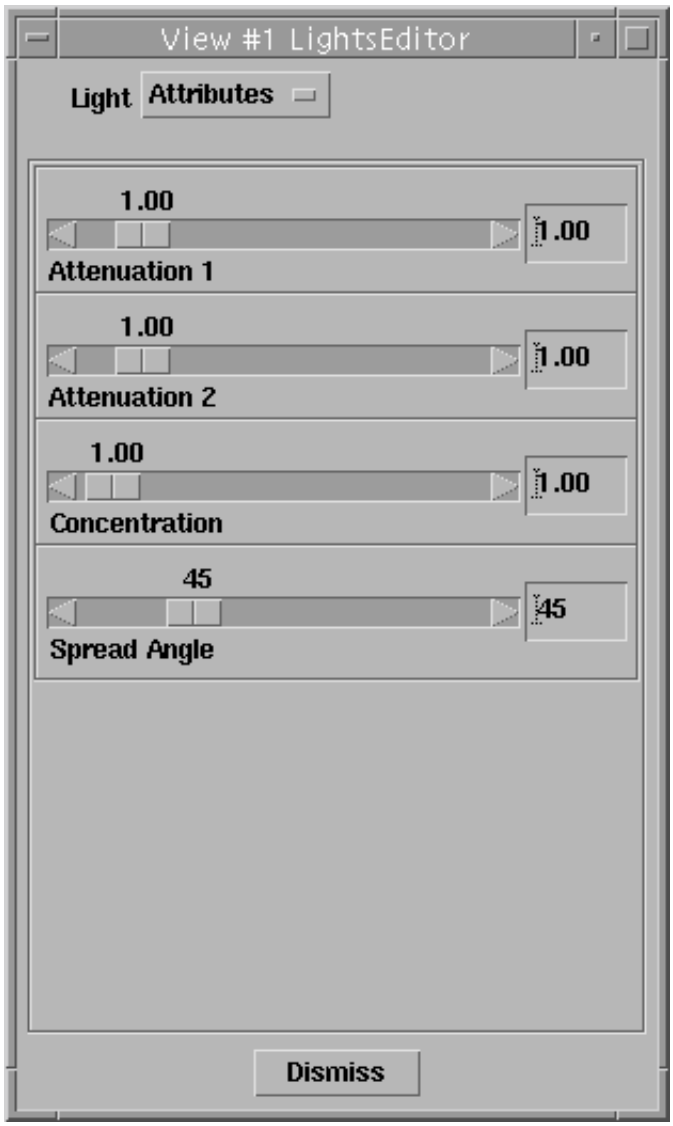

View #1 CameraEditor
Camera
Lens
Perspective On
Global Scale
1.00
Viewport Size
5.00
45
35
Field of View
Dismiss

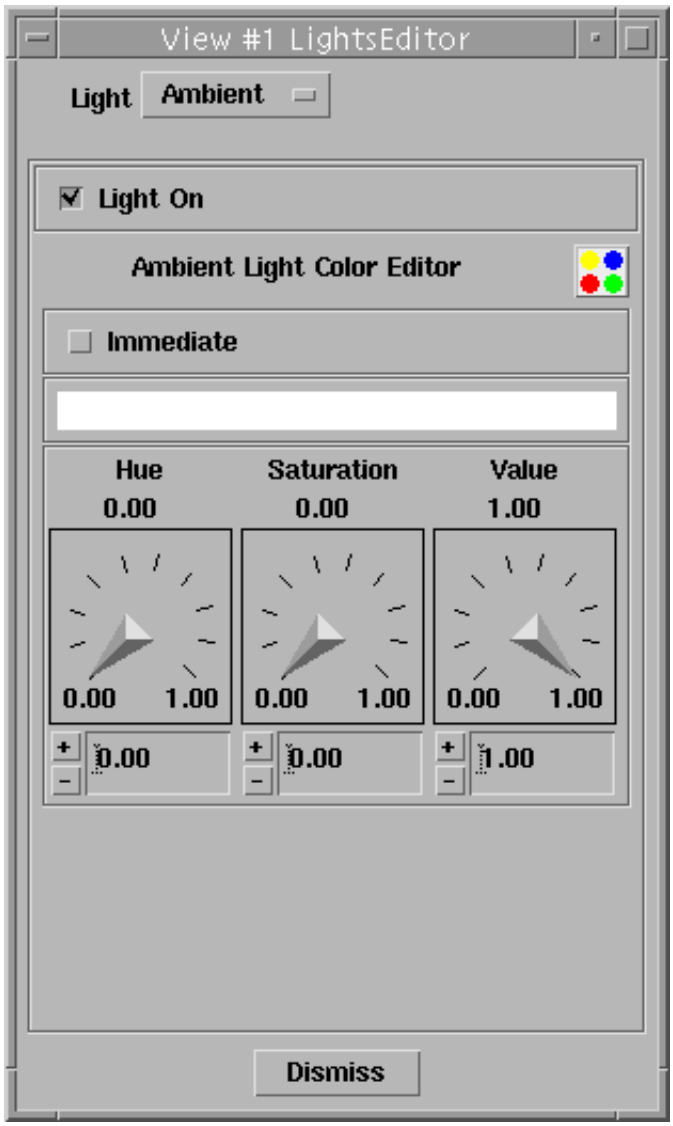

View #1 CameraEditor
Camera
Clipping Planes
Clipping Planes On
Clip Plane Positions
-87.99
Front
112.00
Back
Dismiss


View #1 CameraEditor
Camera
Depth Cueing
Depth Cueing On
9.00
9.00
Front Depth Cue
15.00
15.00
Back Depth Cue
.10
0.10
Depth Cue Scale
Dismiss

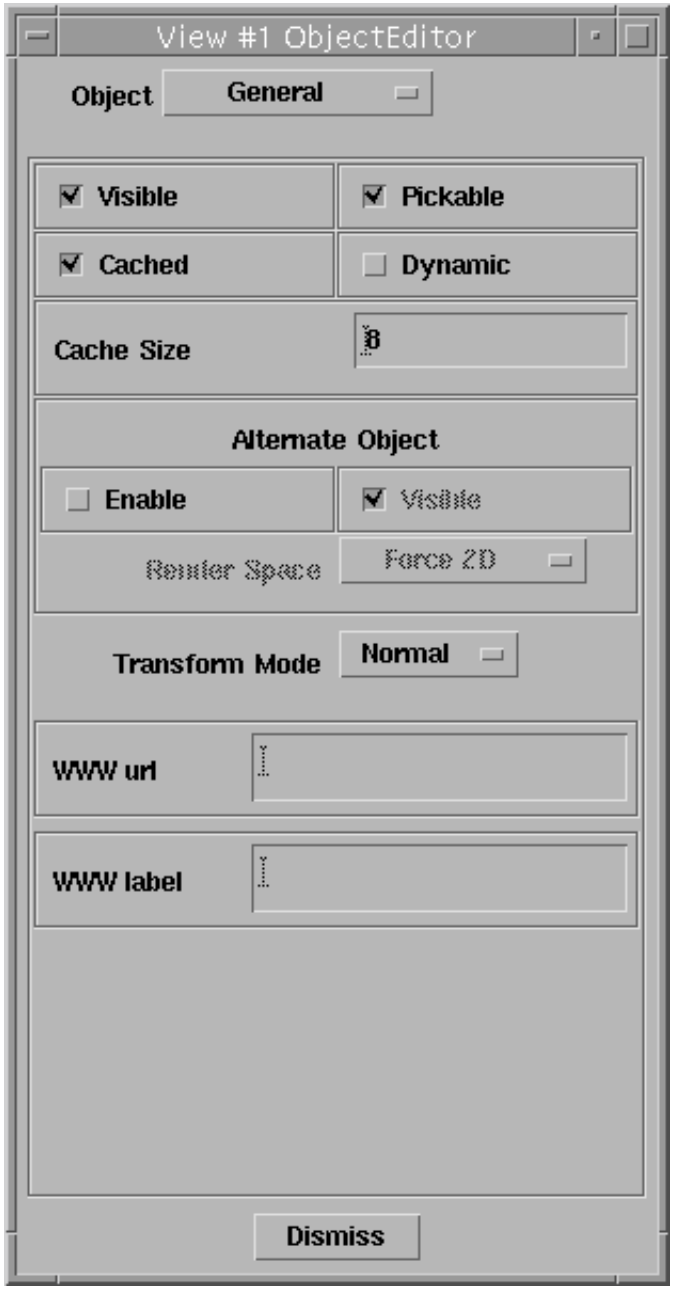
6.3.3 Lights Editors

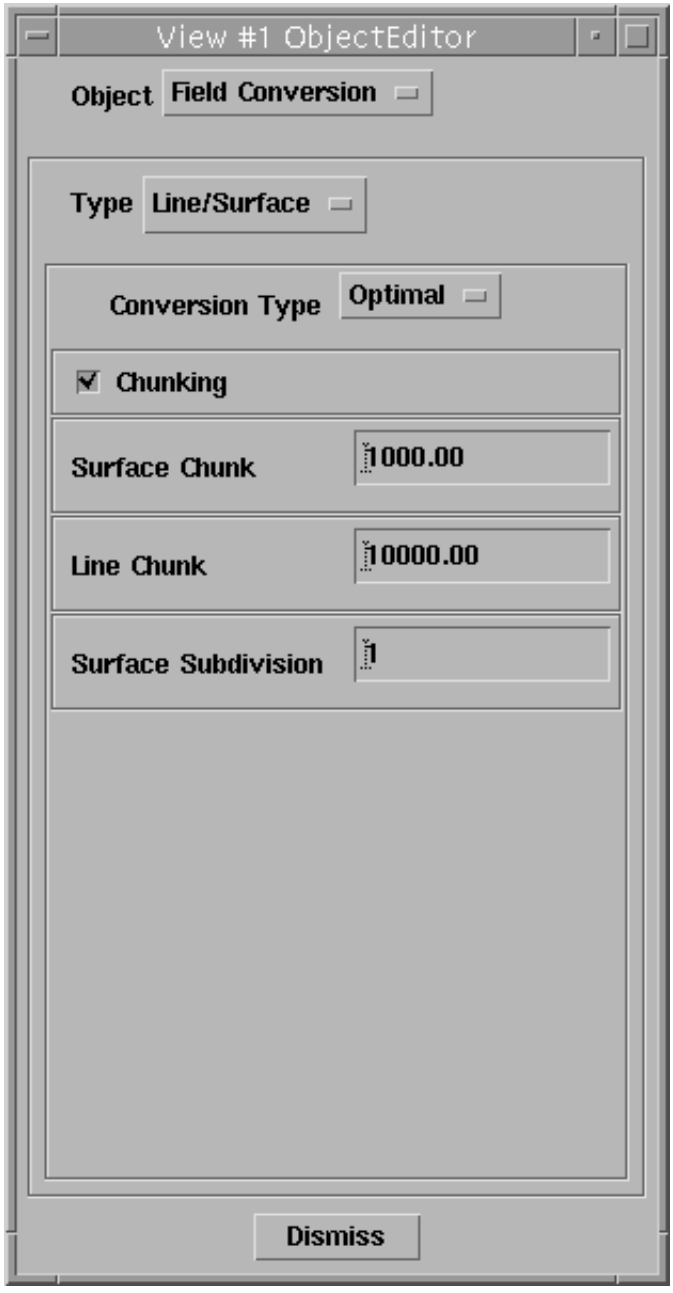

View #1 LightsEditor
Light
General
Current Light
Light 0
Light Type
Directional
Light On
Reset
Current Light Color Editor
Immediate
Hue
Saturation
Value
0.00
0.00
1.00
0.00
1.00
0.00
1.00
0.00
1.00
0.00
1.00
Dismiss

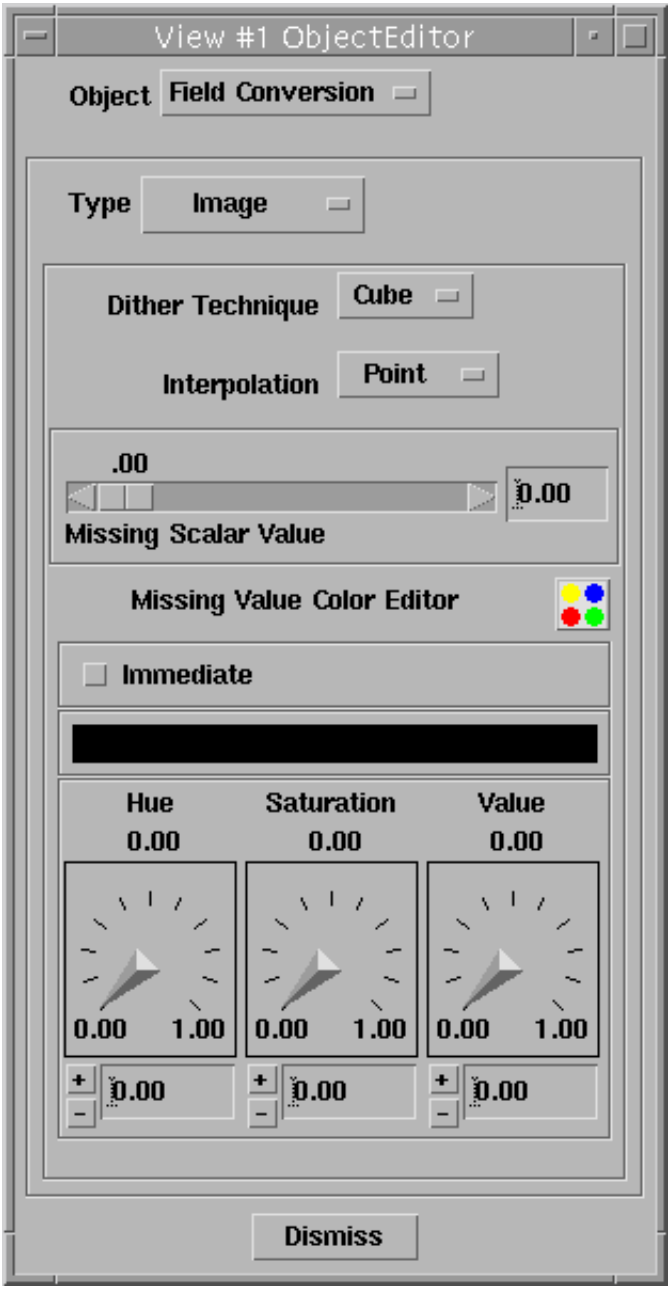

View #1 LightsEditor
Light
Attributes
1.00
1.00
Attenuation 1
1.00
1.00
Attenuation 2
1.00
1.00
Concentration
45
45
Spread Angle
Dismiss

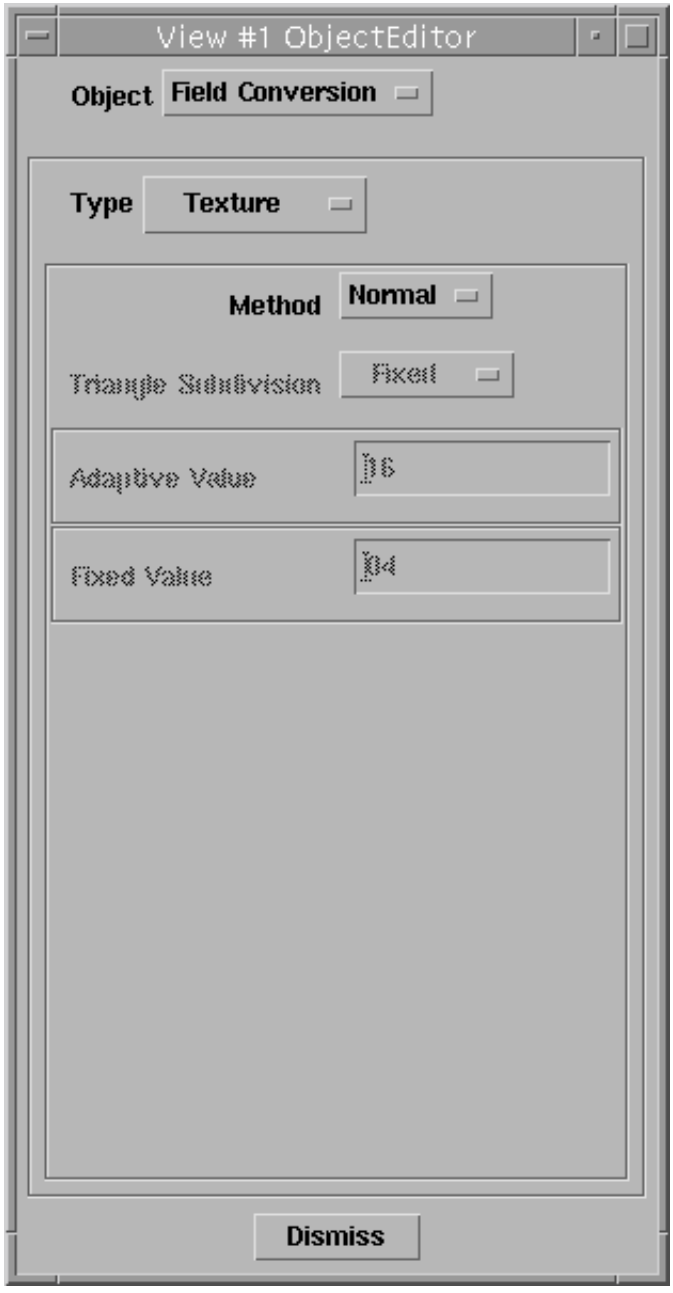

View #1 LightsEditor
Light
Ambient
Light On
Ambient Light Color Editor
Immediate
Hue
Saturation
Value
0.00
0.00
1.00
0.00
1.00
0.00
1.00
0.00
1.00
0.00
1.00
Dismiss

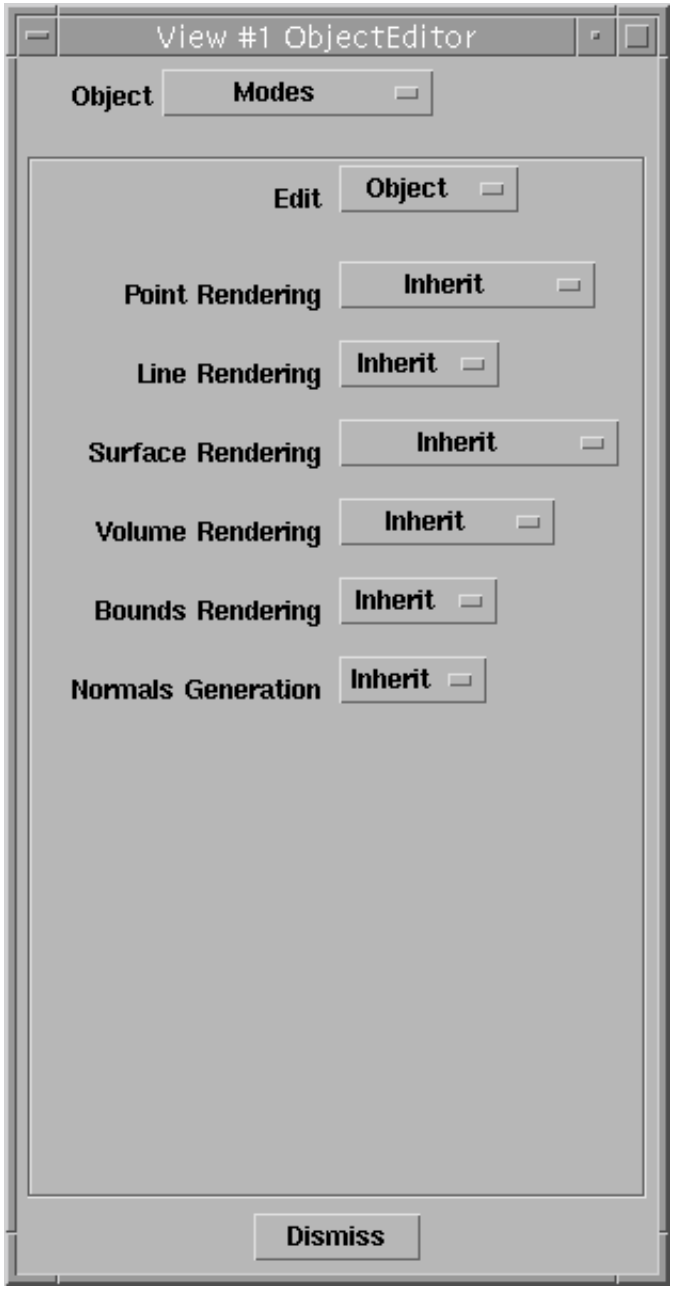
6.3.4 Object Editor

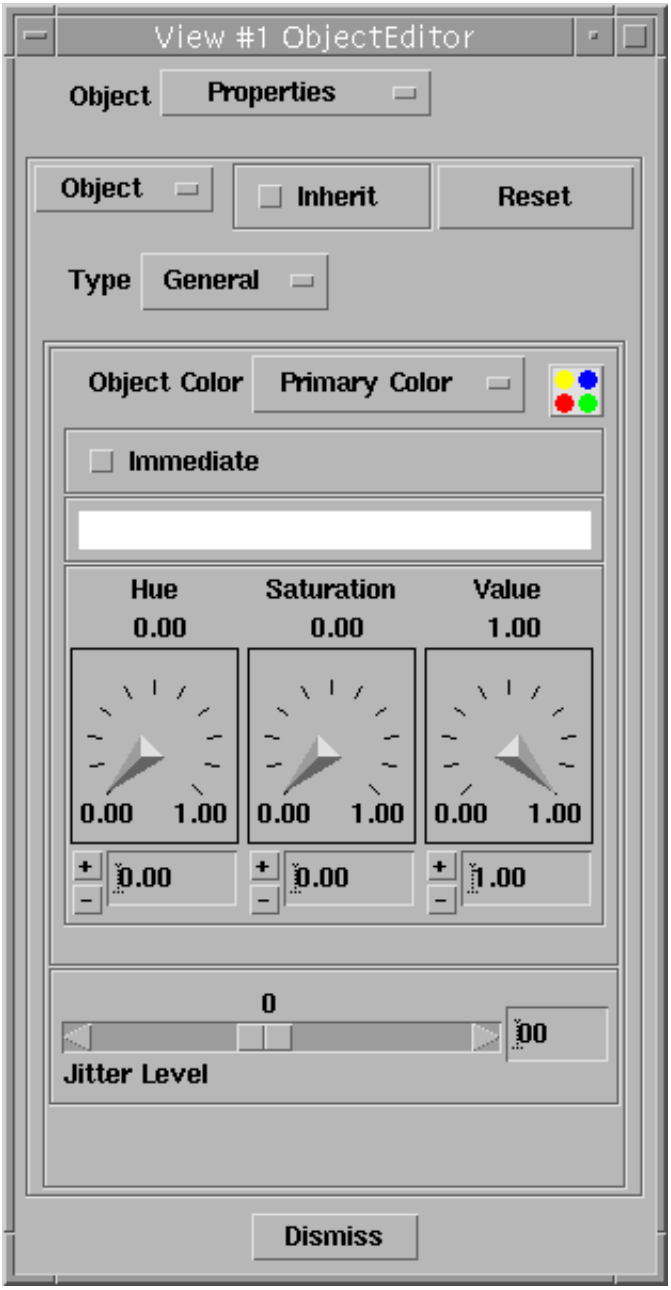

View #1 ObjectEditor
Object
General
Visible
Pickable
Cached
Dynamic
Cache Size
8
Alternate Object
Enable
Visible
Render Space
Force 2D
Transform Mode
Normal
WWW url
WWW label
Dismiss


View #1 ObjectEditor
Object
Field Conversion
Type
Line/Surface
Conversion Type
Optimal
Chunking
Surface Chunk
1000.00
Line Chunk
10000.00
Surface Subdivision
1
Dismiss

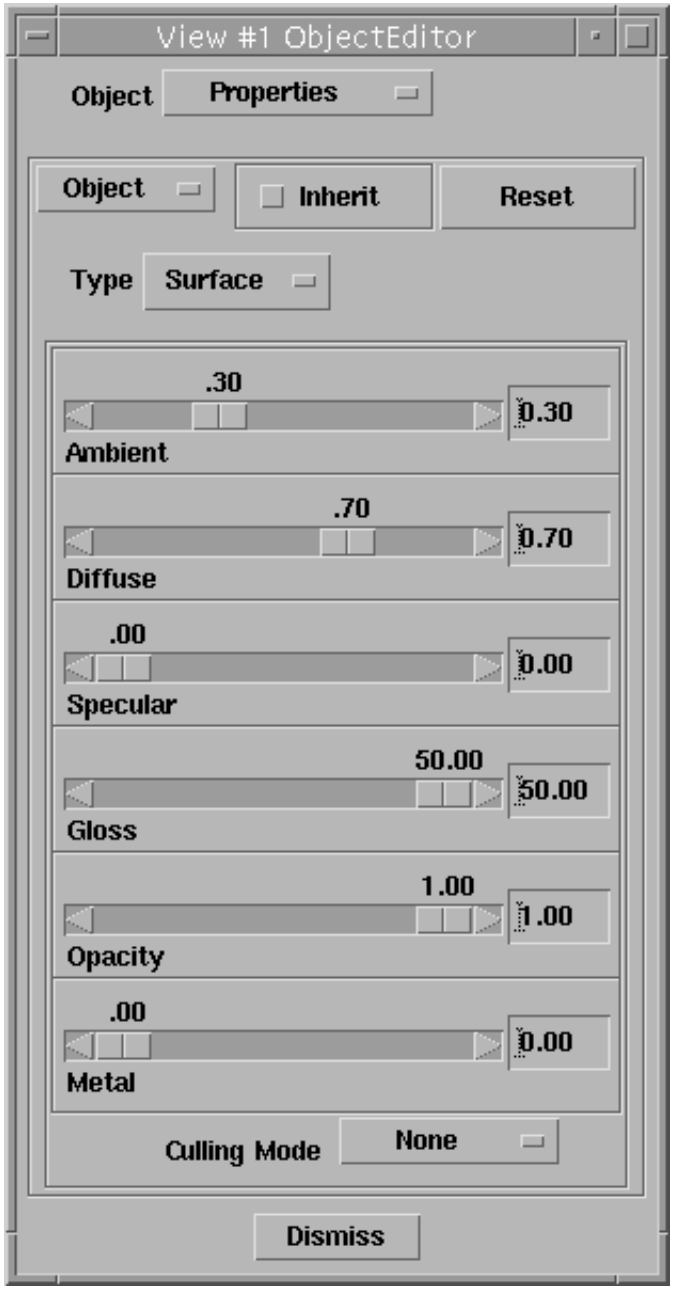

View #1 ObjectEditor
Object
Field Conversion
Type
Image
Dither Technique
Cube
Interpolation
Point
.00
0.00
Missing Scalar Value
Missing Value Color Editor
Immediate
Hue
Saturation
Value
0.00
0.00
0.00
0.00
1.00
0.00
1.00
0.00
1.00
0.00
0.00
0.00
Dismiss

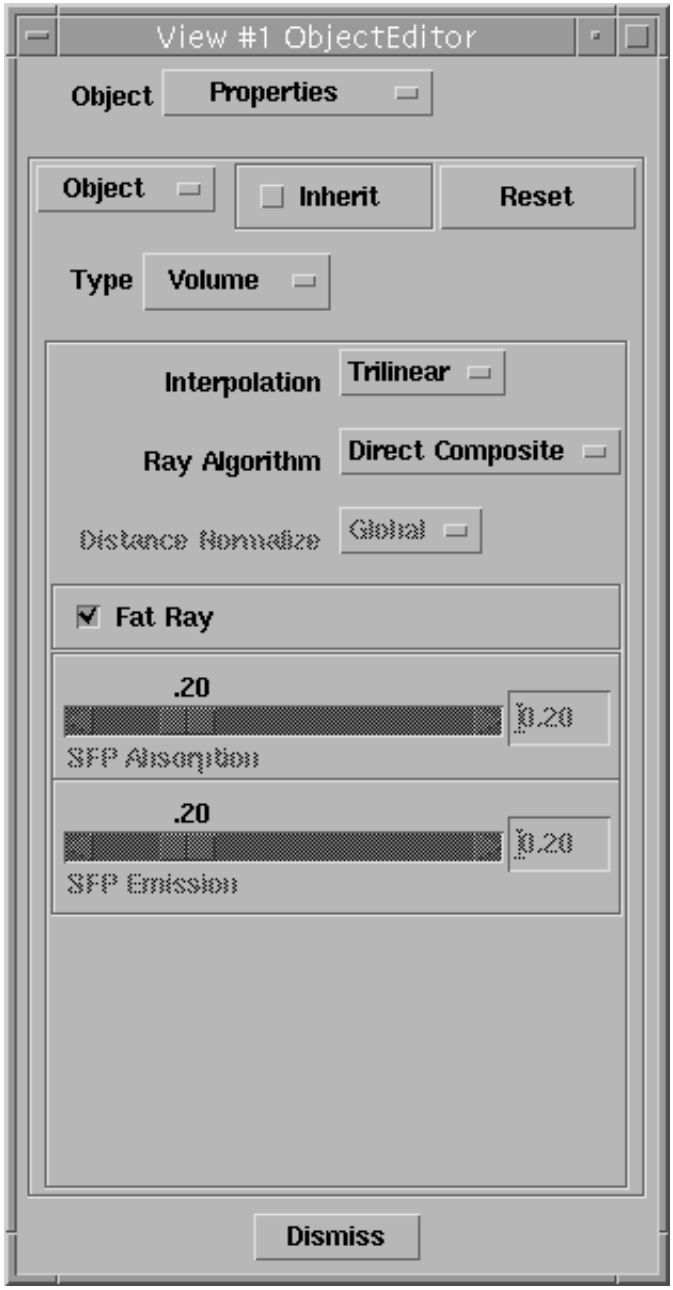

View #1 ObjectEditor
Object
Field Conversion
Type
Texture
Method
Normal
Triangle Subdivision
Fixed
Adaptive Value
16
Fixed Value
4
Dismiss


View #1 ObjectEditor
Object
Modes
Edit
Object
Point Rendering
Inherit
Line Rendering
Inherit
Surface Rendering
Inherit
Volume Rendering
Inherit
Bounds Rendering
Inherit
Normals Generation
Inherit
Dismiss

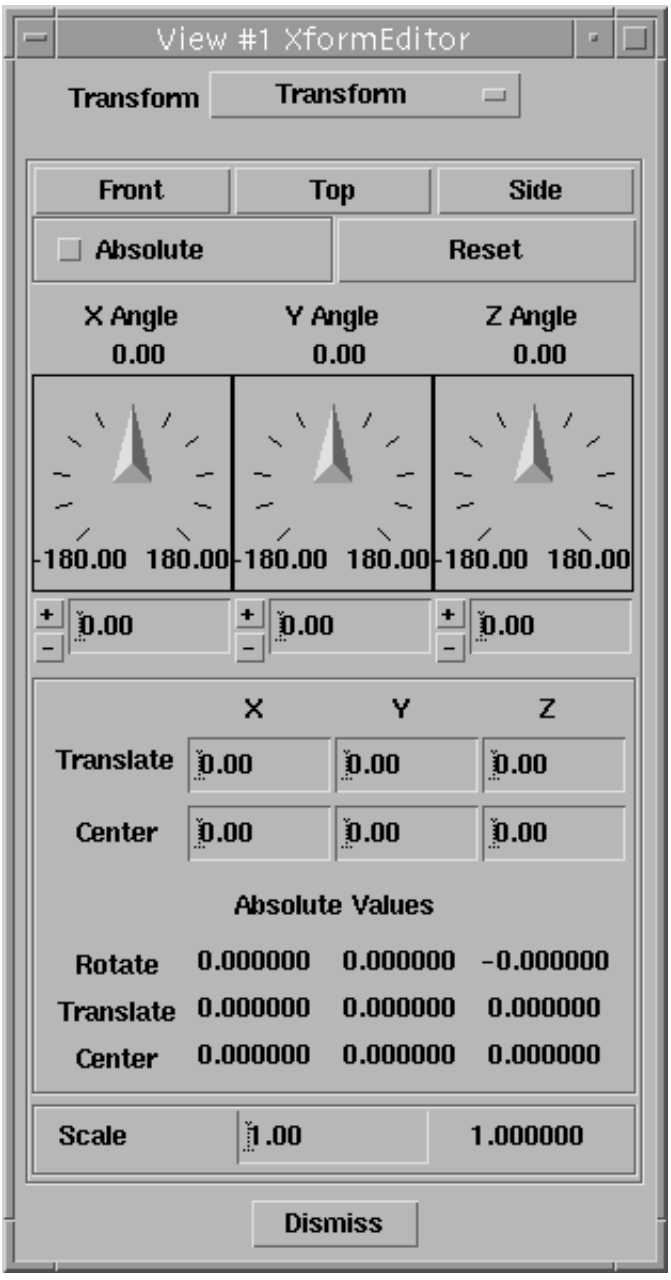

View #1 ObjectEditor
Object
Properties
Object
Inherit
Reset
Type
General
Object Color
Primary Color
Immediate
Hue
Saturation
Value
0.00
0.00
1.00
0.00
1.00
0.00
1.00
0.00
1.00
0.00
1.00
0
00
Jitter Level
Dismiss

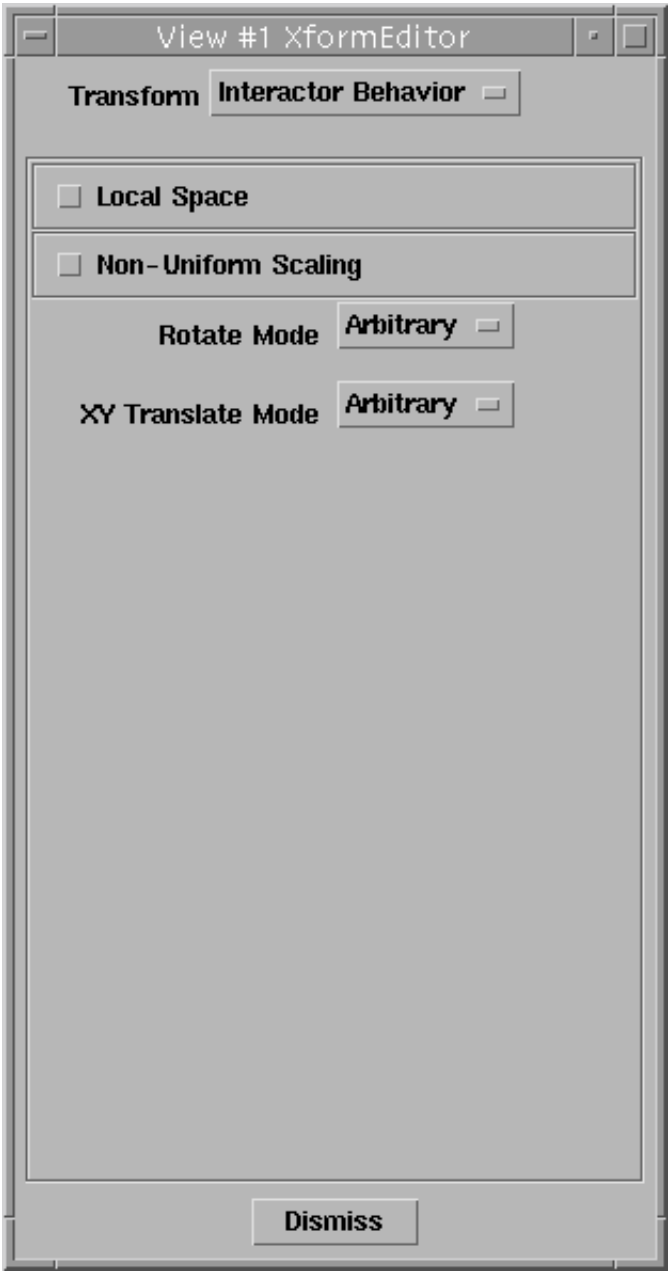

View #1 ObjectEditor
Object
Properties
Object
Inherit
Reset
Type
Point/Line
Line Style
Solid
Drawing Mode
Copy
0
00
Line Thickness
.50
0.50
Glyph Size
4
04
Subdivision
Smooth lines
Dismiss


View #1 ObjectEditor
Object
Properties
Object
Inherit
Reset
Type
Surface
.30
0.30
Ambient
.70
0.70
Diffuse
.00
0.00
Specular
50.00
50.00
Gloss
1.00
1.00
Opacity
.00
0.00
Metal
Culling Mode
None
Dismiss


View #1 ObjectEditor
Object
Properties
Object
Inherit
Reset
Type
Volume
Interpolation
Trilinear
Ray Algorithm
Direct Composite
Distance Normalize
Global
Fat Ray
.20
0.20
SFP Absorption
.20
0.20
SFP Emission
Dismiss

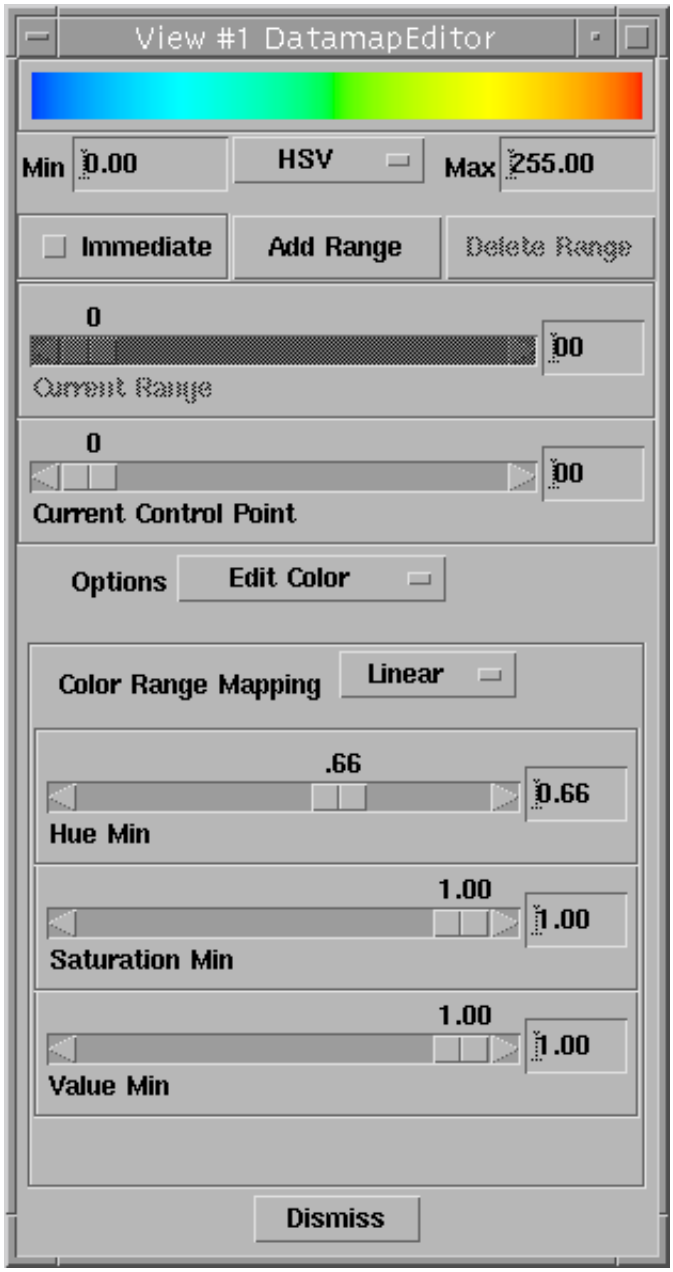
6.3.5 Transform Editor

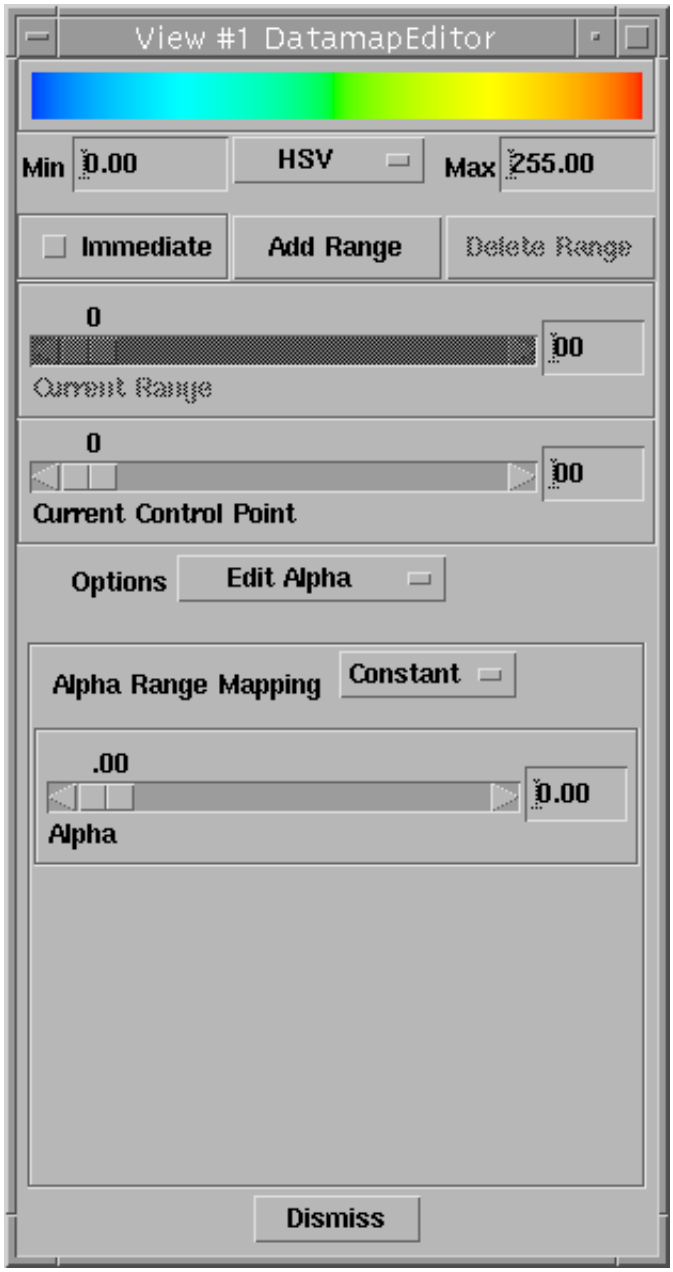

View #1 XformEditor
Transform
Transform
Front
Top
Side
Absolute
Reset
X Angle
0.00
Y Angle
0.00
Z Angle
0.00
-180.00
180.00
-180.00
180.00
-180.00
180.00
0.00
0.00
0.00
X
Y
Z
Translate
0.00
0.00
0.00
Center
0.00
0.00
0.00
Absolute Values
Rotate
0.000000
0.000000
-0.000000
Translate
0.000000
0.000000
0.000000
Center
0.000000
0.000000
0.000000
Scale
1.00
1.000000
Dismiss

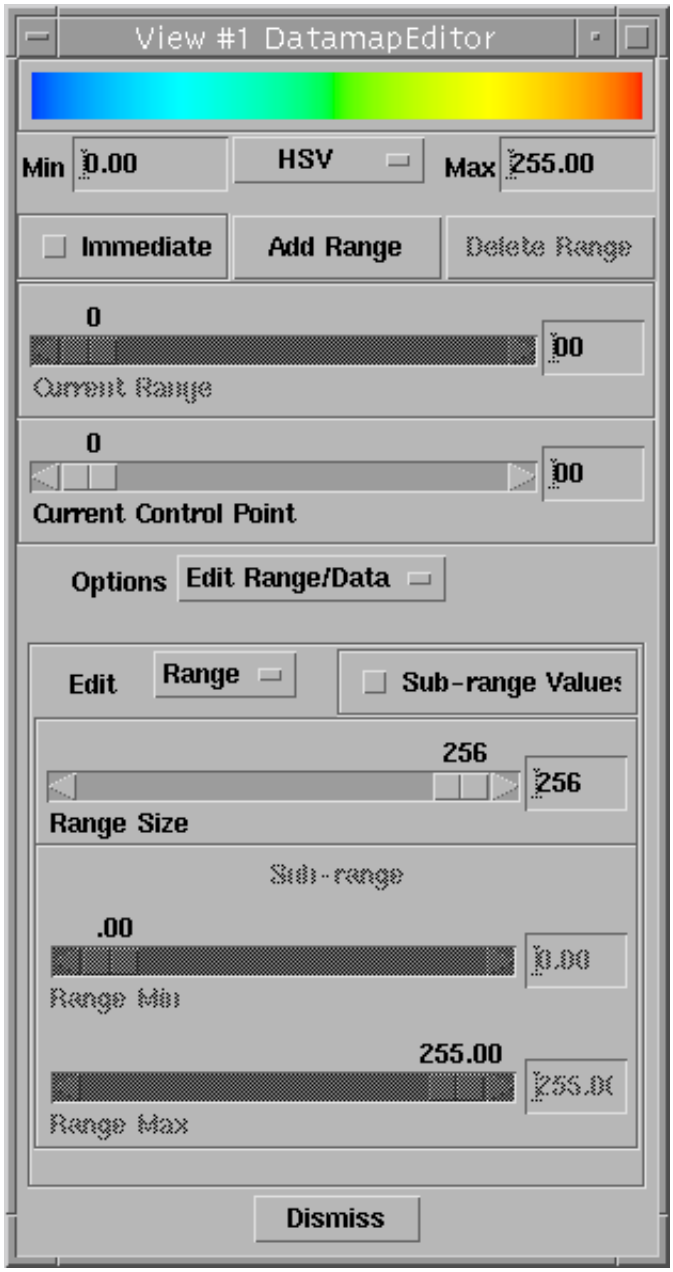

View #1 XformEditor
Transform
Interactor Behavior
Local Space
Non-Uniform Scaling
Rotate Mode
Arbitrary
XY Translate Mode
Arbitrary
Dismiss

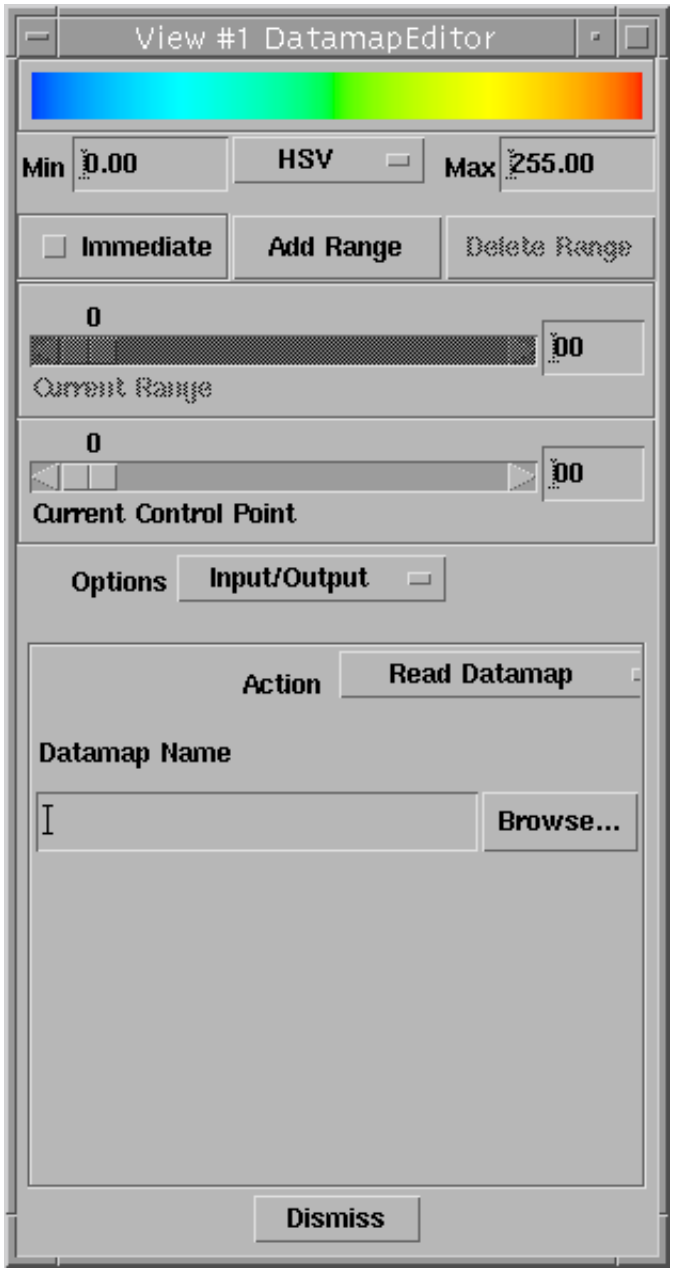
6.3.6 Datamap Editor

This is the popup window for the datamap editor that is available in some of the visualization methods. It is used to control the data-to-color mapping. Full discussion of its functionality is available in the AVS/Express documentation set so only a brief discussion will given here. The Hue, Saturation, and Value slider widgets are used to values at the top and bottom of the data scale shown at the left. Color values are then linearly interpolated between the two values. Alternately, a RGB color model can also be selected. When *Range* is selected in the *Edit Selection* box, the window will present the above view of the datamap editor. The *Min Range Value* and *Max Range Value* widgets specify the data range will be mapped to the specified colormap. The *Min Range Value* must be greater than or equal to the minimum data value in the data field and the *Max Range Value* must be less than or equal to the maximum data value in the data field. The *Range Value* toggle widget controls whether or not these new values are used or if the minimum and maximum values in the data field are used. The *Range Size* typein sets the number of color levels that will be computed.

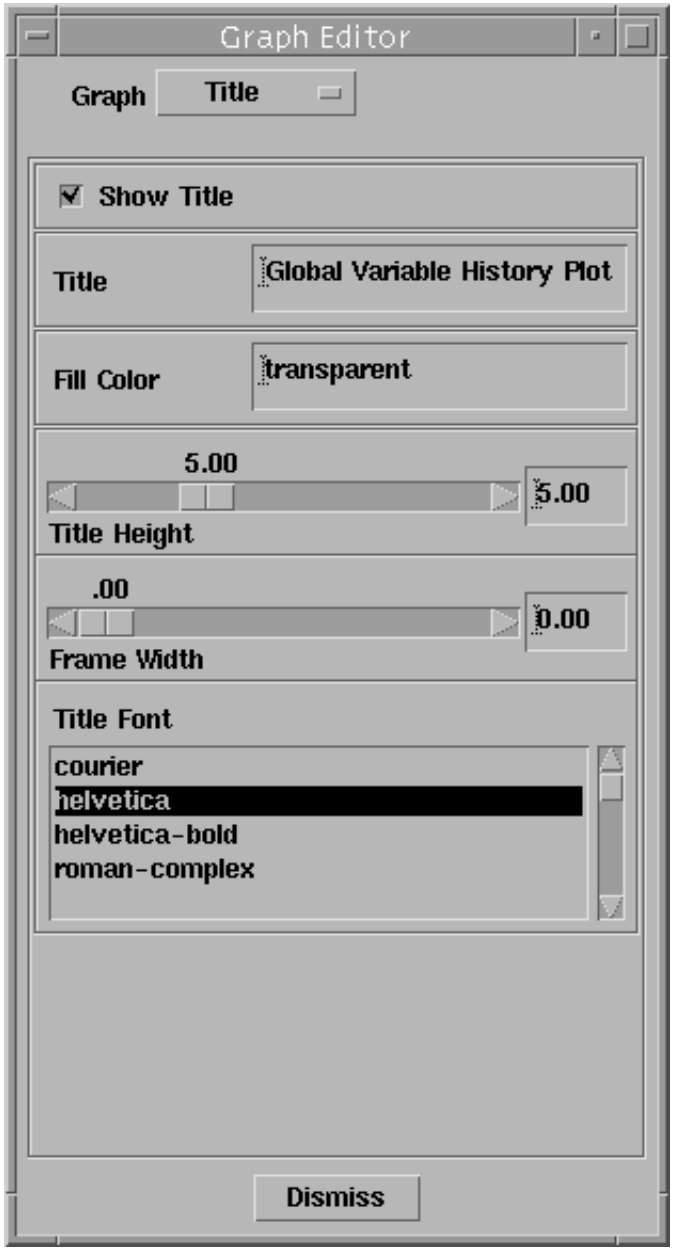

View #1 DatamapEditor
Min
0.00
HSV
Max
255.00
Immediate
Add Range
Delete Range
0
00
Current Range
0
00
Current Control Point
Options
Edit Color
Color Range Mapping
Linear
.66
0.66
Hue Min
1.00
1.00
Saturation Min
1.00
1.00
Value Min
Dismiss

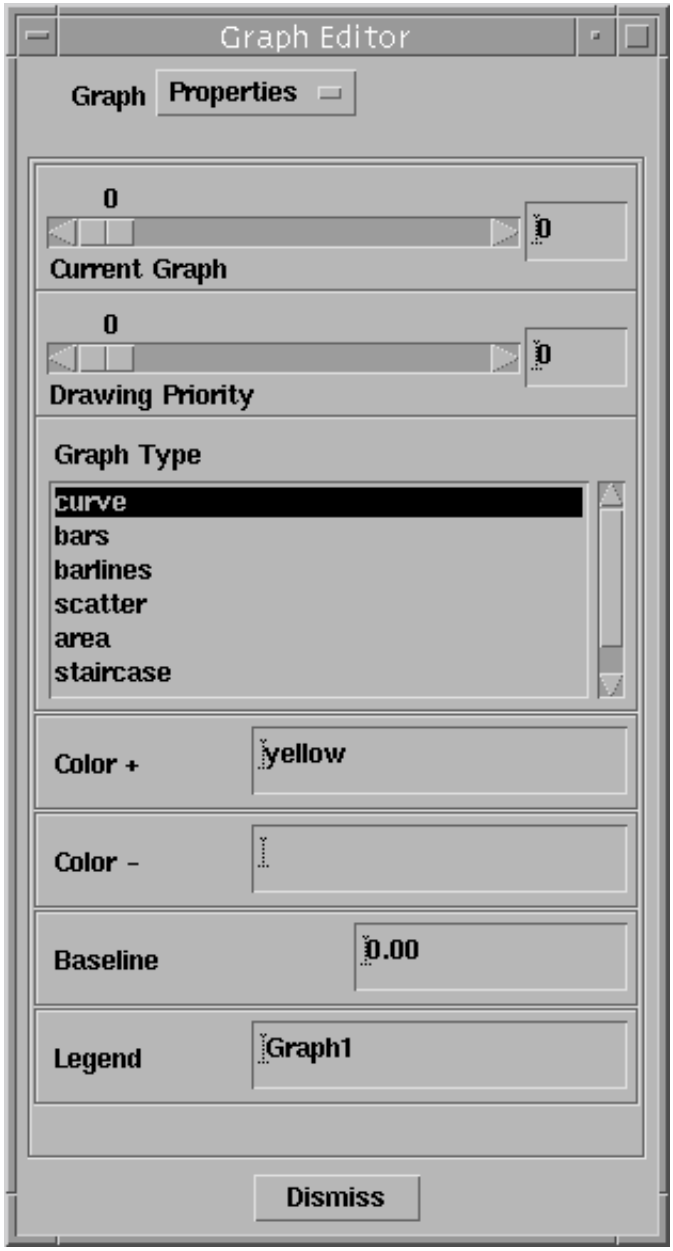

View #1 DatamapEditor
Min
0.00
HSV
Max
255.00
Immediate
Add Range
Delete Range
0
Current Range
0
Current Control Point
Options
Edit Alpha
Alpha Range Mapping
Constant
.00
0.00
Alpha
Dismiss

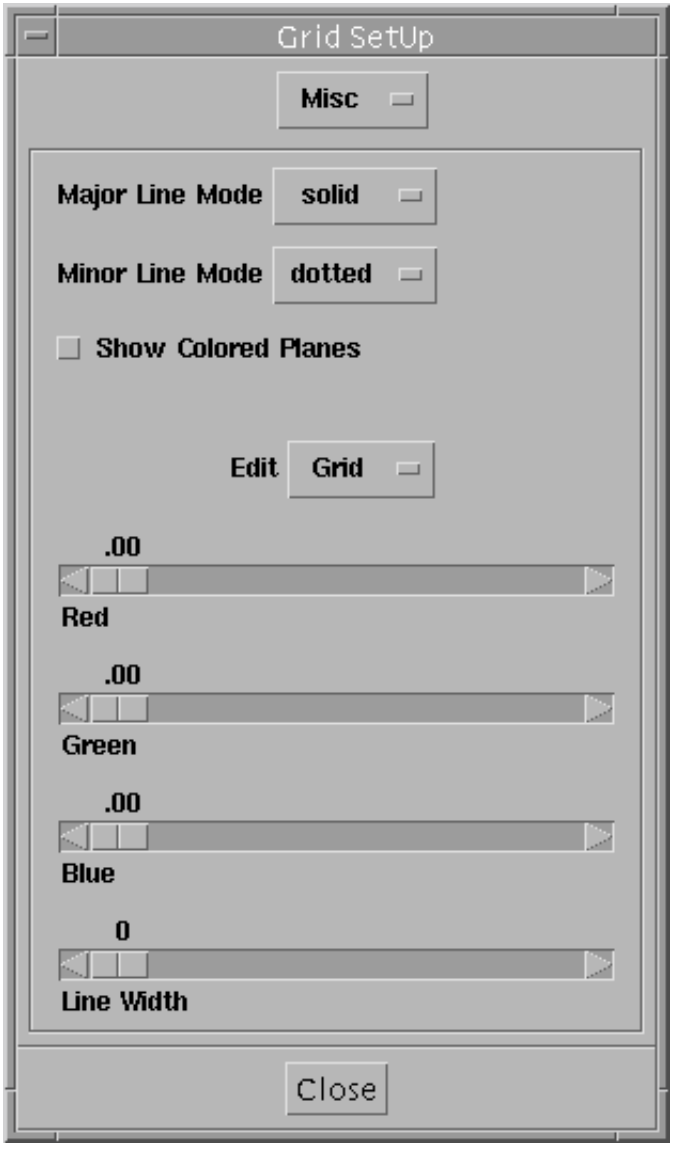

View #1 DatamapEditor
Min
0.00
HSV
Max
255.00
Immediate
Add Range
Delete Range
0
Current Range
0
Current Control Point
Options
Edit Range/Data
Edit
Range
Sub-range Values
256
Range Size
Sub-range
.00
Range Min
255.00
Range Max
Dismiss

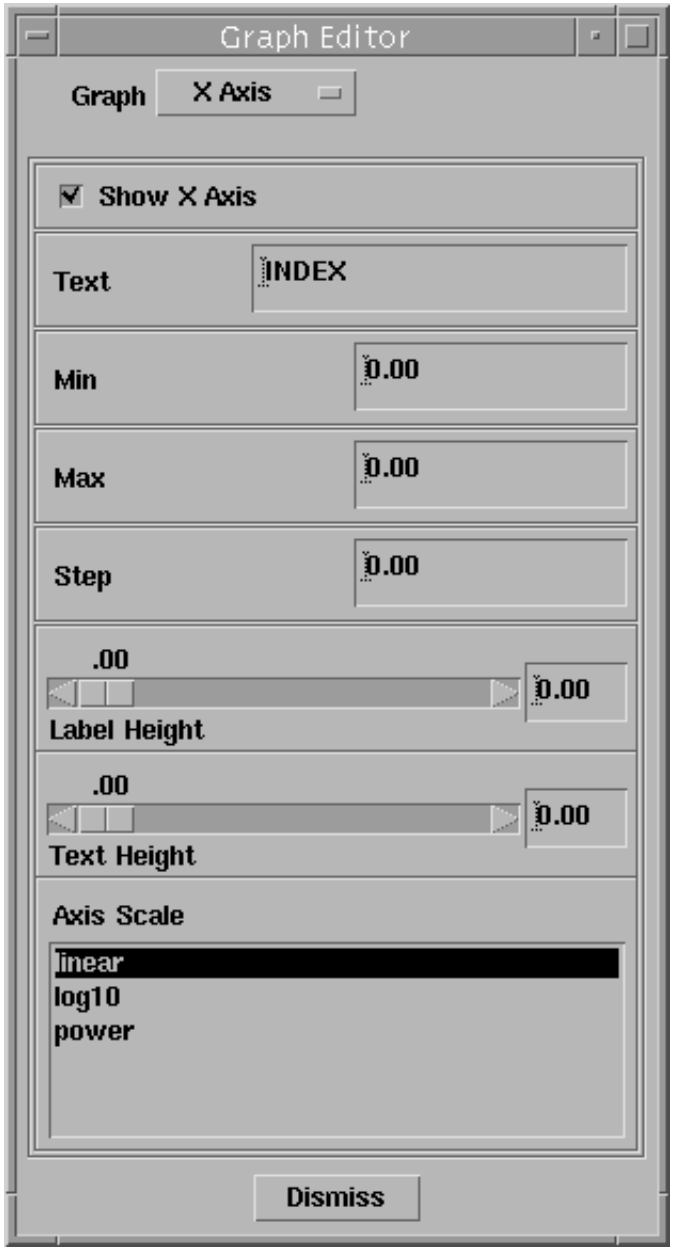

View #1 DatamapEditor
Min
0.00
HSV
Max
255.00
Immediate
Add Range
Delete Range
0
Current Range
0
Current Control Point
Options
Input/Output
Action
Read Datamap
Datamap Name
Browse...
Dismiss

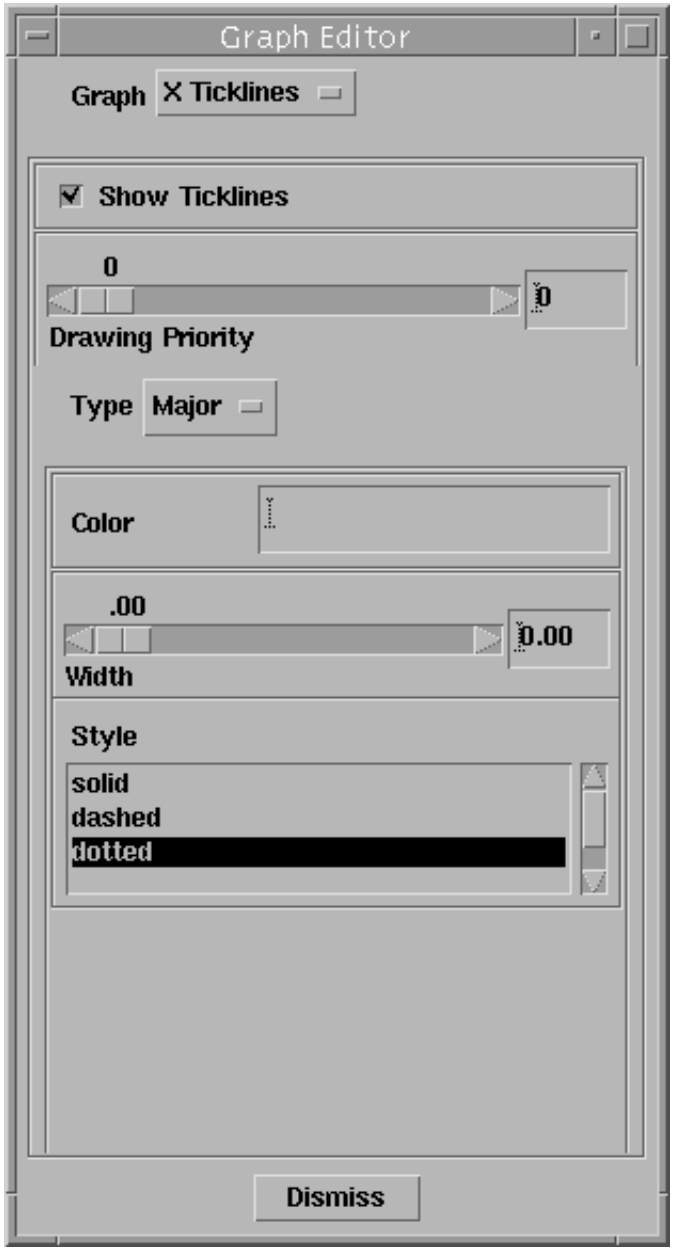
6.3.7 Graph Editor


Graph Editor
Graph
Title
Show Title
Title
Global Variable History Plot
Fill Color
transparent
5.00
5.00
Title Height
.00
0.00
Frame Width
Title Font
courier
helvetica
helvetica-bold
roman-complex
Dismiss

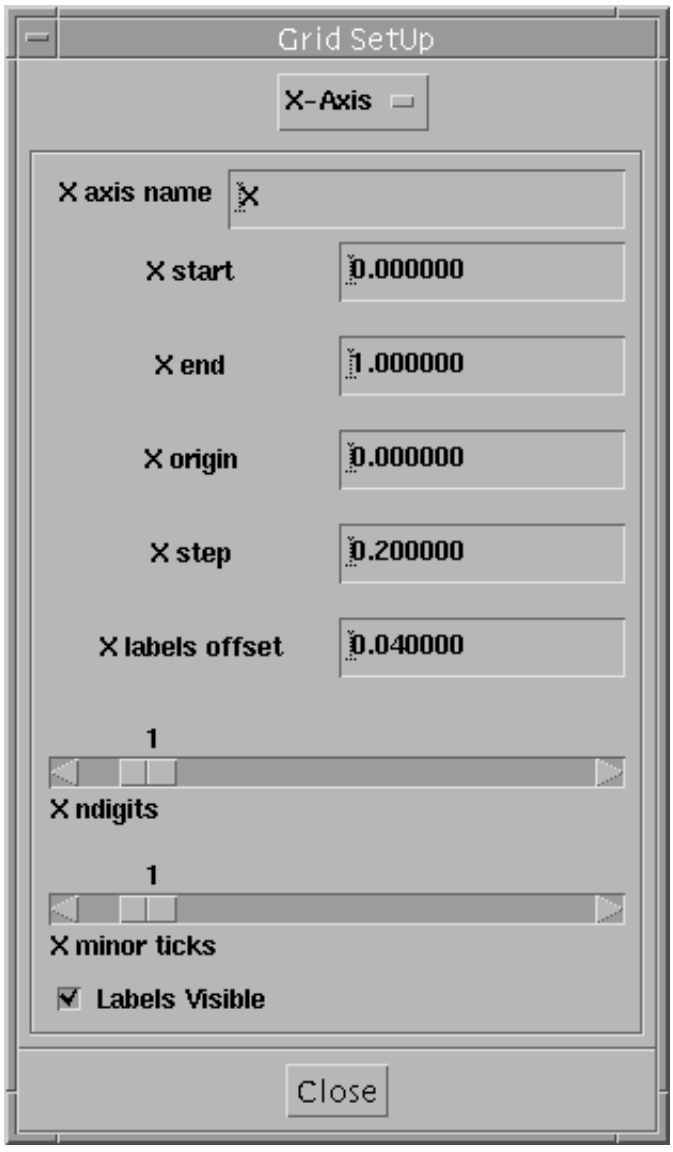

Graph Editor
Graph
Properties
0
0
Current Graph
0
0
Drawing Priority
Graph Type
curve
bars
barlines
scatter
area
staircase
Color +
yellow
Color -
Baseline
0.00
Legend
Graph1
Dismiss


Grid SetUp
Misc
Major Line Mode
solid
Minor Line Mode
dotted
Show Colored Planes
Edit
Grid
.00
Red
.00
Green
.00
Blue
0
Line Width
Close


Graph Editor
Graph
X Axis
Show X Axis
Text
INDEX
Min
0.00
Max
0.00
Step
0.00
.00
0.00
Label Height
.00
0.00
Text Height
Axis Scale
linear
log10
power
Dismiss


Graph Editor
Graph
X Ticklines
Show Ticklines
0
0
Drawing Priority
Type
Major
Color
.00
0.00
Width
Style
solid
dashed
dotted
Dismiss

6.3.8 Grid Editor


Grid SetUp
X-Axis
X axis name
X
X start
0.000000
X end
1.000000
X origin
0.000000
X step
0.200000
X labels offset
0.040000
1
X ndigits
1
X minor ticks
Labels Visible
Close

7. Miscellanious

7.1 Play Image Sequence

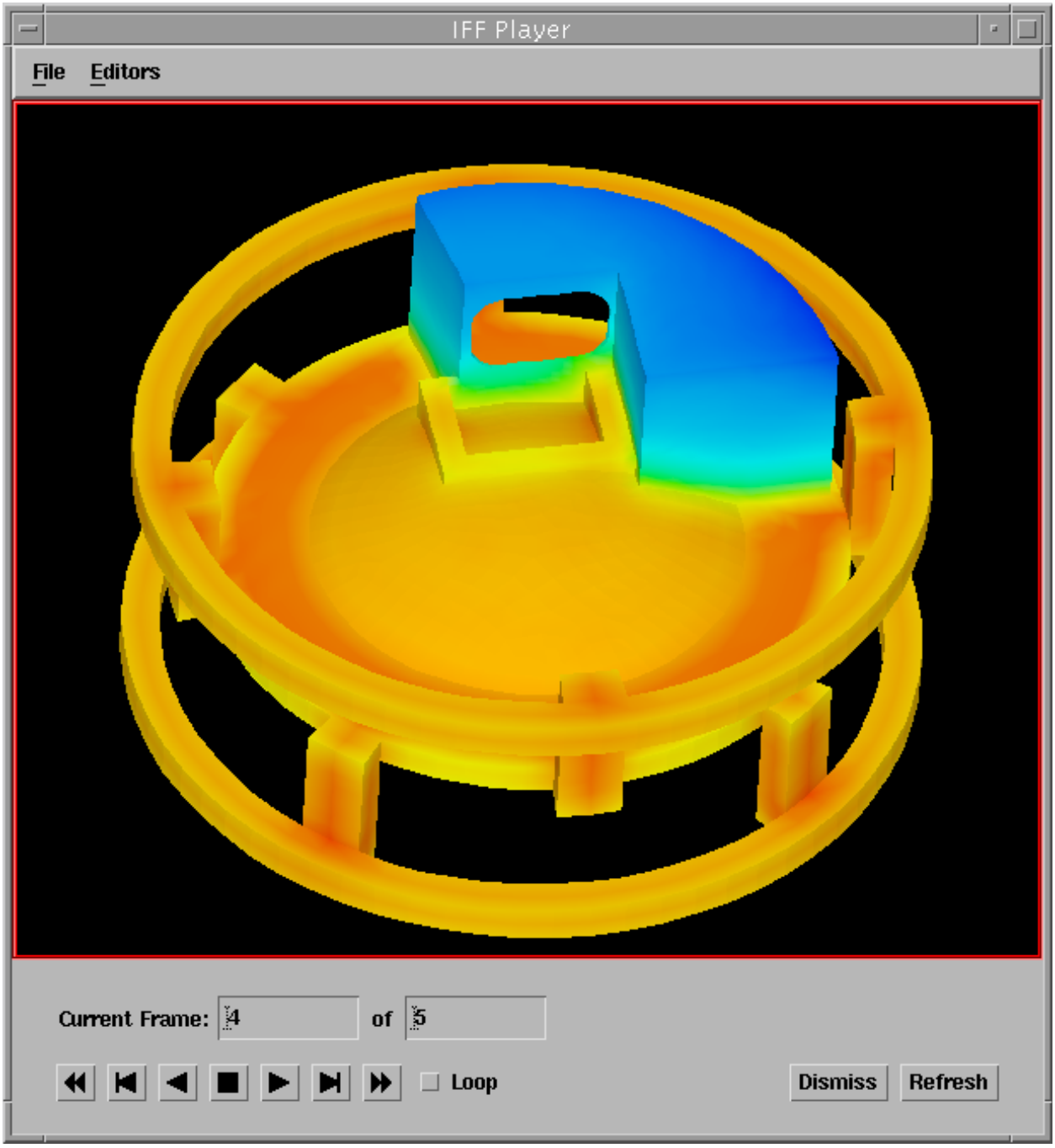


Figure 6.1

7.2 Record Video

This modules provides an interface to Sony's laser video disc recorder. The interface can be either via a direct TTY serial port connection or via sockets using the network client/server package I wrote a few years ago.

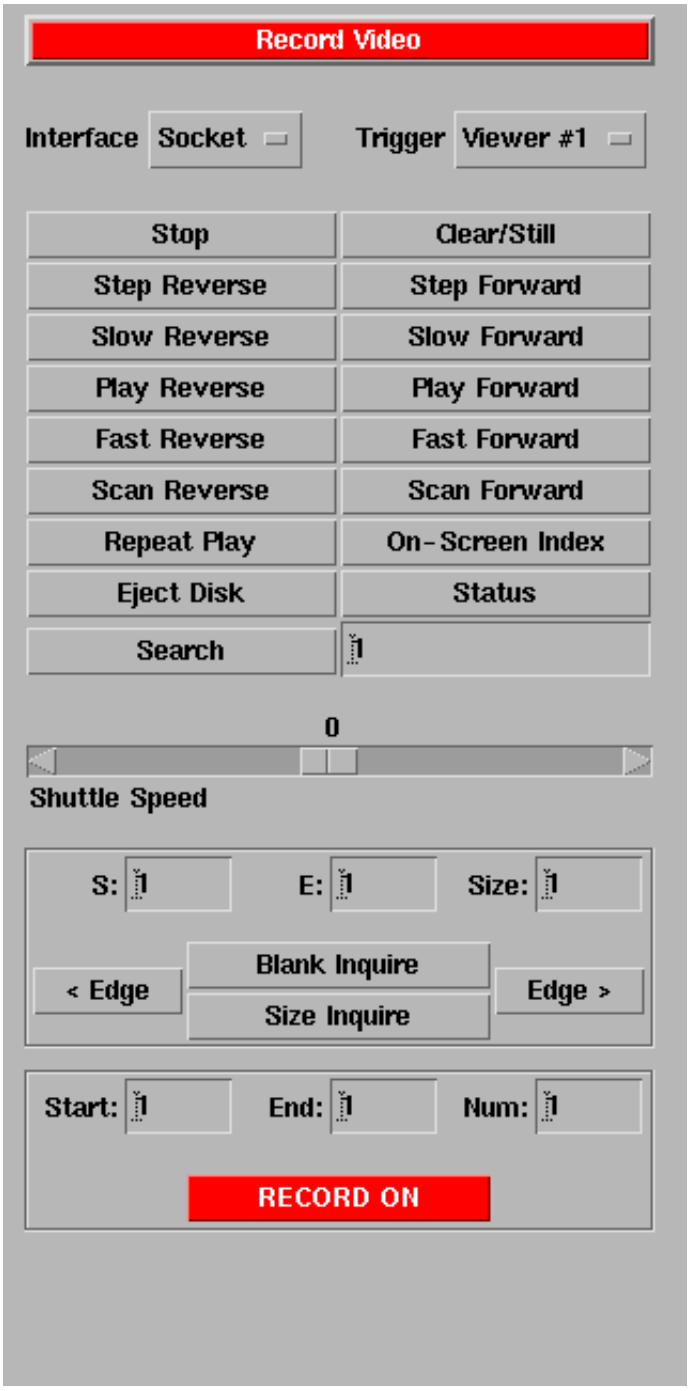
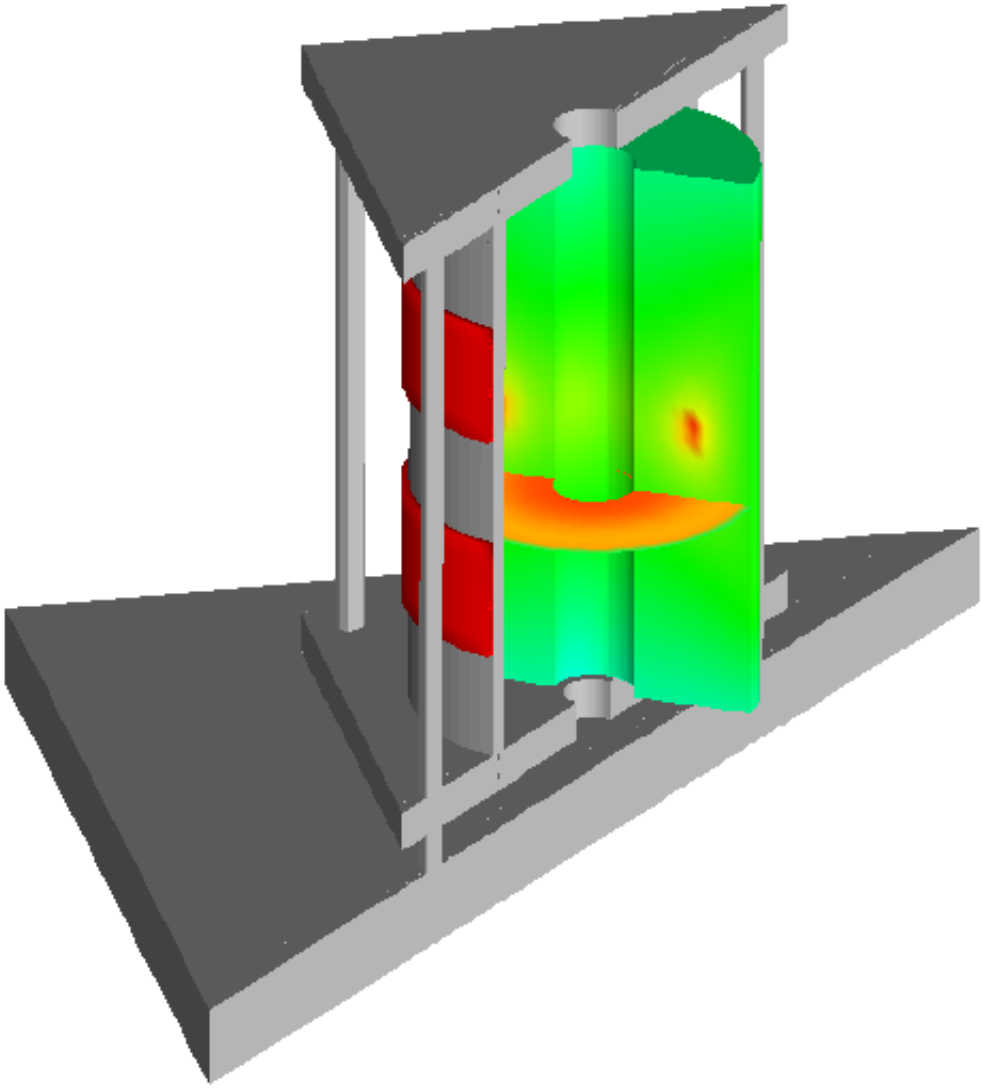
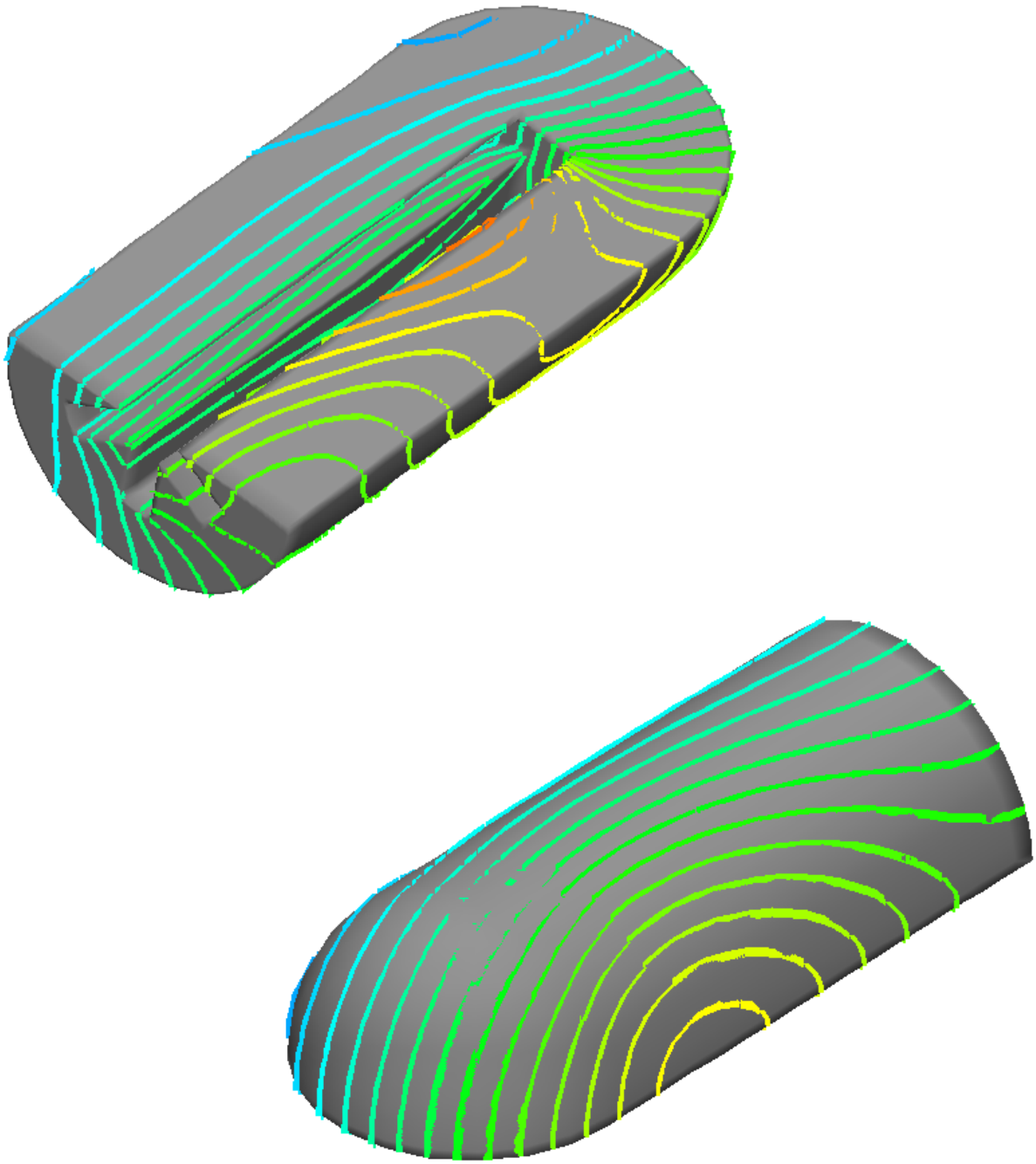


Figure 6.2 Control panel for "Record Video"

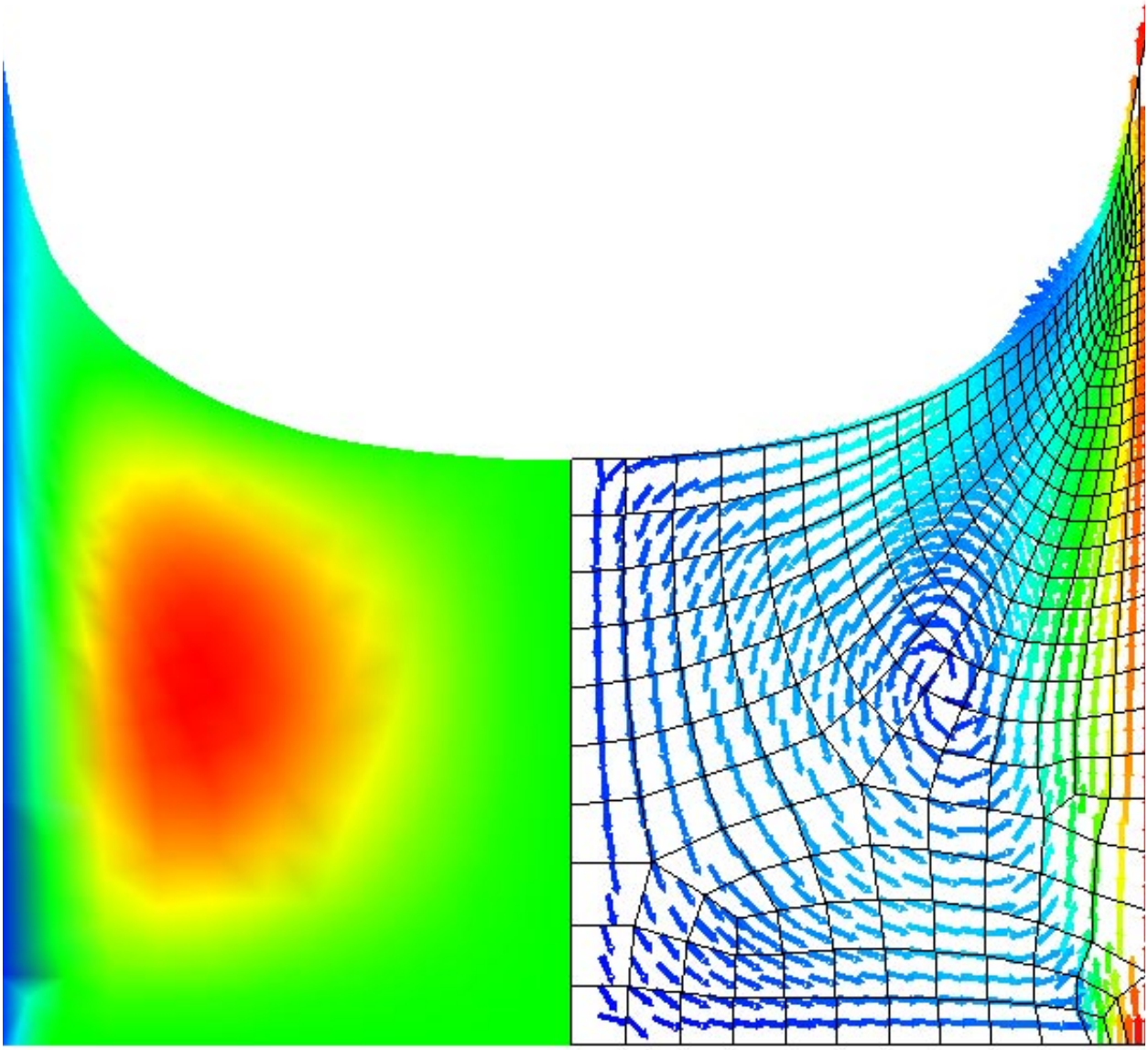
8. Sample Visualizations



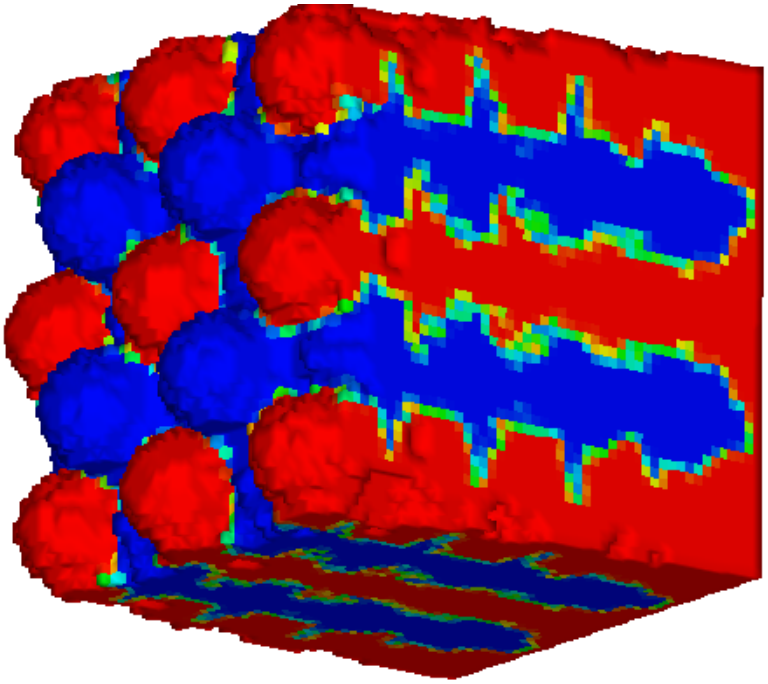
This image was produced from an ExodusII dataset containing three different element blocks. It is comprised of four visualization methods with local filters used to select different element blocks for each method. First, a *Cut Away* method was used on only the element block representing the metal casing. The colormap was disabled and a neutral gray color was assigned. Second, another *Cut Away* method was used on only the element block representing the heating bands. The colormap was disabled and a red color was assigned. Thirdly, another *Cut Away* method was used on only the element block representing the energetic material and temperature is colormapped onto its surface. And finally, a *Slice Plane* method was used on only the element block representing the energetic material and the species concentration is colormapped onto it



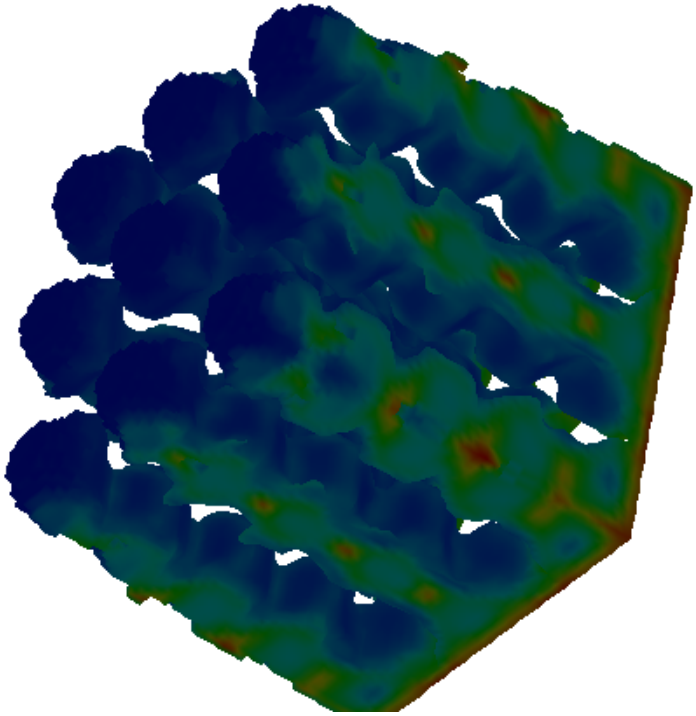
This image was produced from an ExodusII dataset containing three different element blocks. It is comprised of two visualization methods and a filter. The filter is used to remove two of the three element blocks. Then a *Show Mesh* method was used to display the model in solid mode and a neutral gray color assigned to it. Finally, an *Iso Lines* method was used to display temperature iso lines on the model surface.



This image was produced from an ExodusII dataset containing one element block. It is comprised of three visualization methods and a global *Mesh Displacements* filter. The first visualization method is an *External Faces* method that has a local *Mirror* filter applied to reflect the model about the y-axis. Secondly, a *Show Mesh* method was used to display the model in hidden surface wireframe mode. Finally, a *HedgeHog* method was used to display velocity vectors. The vectors are normalized and colormapped to the velocity magnitude.



This image above was produced from an CTH database that was converted to ExodusII format. The dataset was first thresholded based upon void fraction to remove all the elements with all void space. A "*material*" data value was created from the volume fractions of material 1 and material 2 using the ExodusII reader option and visualized with the *Paint Cells* viz method. The image below was created by using the *Iso Volume* viz method to create an iso volume of material 1 with volume fraction between 0.5 and 1.0 and the color mapping temperature onto the iso volume.



9. Distribution

External Distribution:

Internal Distribution:

MS 0601 1126 H. K. Moffat
MS 9042 8741 V. K. Gabrielson
MS 9043 8743 M. L. Callabresi
MS 0151 9000 G. Yonas
MS 0841 9100 P. J. Hommert
MS 0828 9102 R. D. Skocypec
MS 0833 9103 J. H. Biffle
MS 0828 9104 E. D. Gorham
MS 0826 9111 W. Hermina
MS 0826 9111 K. S. Chen
MS 0826 9111 M. Christon
MS 0826 9111 D. K. Gartling
MS 0826 9111 M. W. Glass (50)
MS 0826 9111 C. E. Hickox
MS 0826 9111 S. N. Kempka
MS 0826 9111 R. R. Rao
MS 0826 9111 P. A. Sackinger
MS 0826 9111 P. R. Schunk
MS 0826 9111 J. A. Schutt
MS 0834 9112 A. C. Ratzel
MS 0834 9112 M. R. Baer
MS 0834 9112 R. J. Gross
MS 0834 9112 M. L. Hobbs
MS 0834 9112 P. L. Hopkins
MS 0834 9112 M. J. Martinez
MS 0835 9113 S. E. Gianoulakis
MS 0835 9113 B. L. Bainbridge
MS 0835 9113 S. P. Burns
MS 0835 9113 R. J. Cochran
MS 0835 9113 D. Dobranich
MS 0835 9113 R. E. Hogan
MS 0835 9113 R. R. Lober
MS 0835 9113 V. J. Romero
MS 0827 9114 R. T. McGrath
MS 0825 9115 W. H. Rutledge
MS 0825 9115 B. Hassan
MS 0836 9116 C. W. Peterson
MS 0443 9117 H. S. Morgan
MS 0437 9118 E. P. Chen
MS 1111 9221 D. R. Gardner
MS 1111 9221 S. A. Hutchinson
MS 1111 9221 A. G. Salinger
MS 1111 9221 J. N. Shadid
MS 1109 9224 K. D. Devine
MS 1109 9224 C. T. Vaughan
MS 0441 9225 P. L. Stanton
MS 0441 9225 C. Pavlokos
MS 0441 9225 L. A. Schoof
MS 0819 9231 J. M. McGlaun
MS 0819 9231 E. S. Hertel
MS 0819 9231 J. S. Peery
MS 9018 8523-2 Central Technical Files
MS 0899 13414 Technical Library (5)
MS 0619 13416 Technical Publications
MS 0100 7613-2 Document Processing for DOE/OSTI (2)